Lotus.

IBM

Notes and Domino Connectivity
A Collection of Examples

Access Enterprise Data and Applications

Using LSX LC

Using LS:DO

Constantin Florea

First Edition (March 2001)

ibm.com/redbooks

Redpaper

Preface

In Lotus Domino R5.0x it is possible to maximize the power of Domino applications by using a range of features to get connected with Relational Databases (for example, Oracle or DB/2)

In respect of this, there are essentially two ways in which users can connect a Domino application to an external data source (RDBM's, file system, etc.):

- LotusScript Extension for Lotus Domino Connectors (LSX LC)
- LotusScript Data Objects (LS:DO)

The purpose of this redpaper is to demonstrate the use of these two techniques in accessing a DB/2 database and to give you a collection of samples encompassing the most important features. It was not my intent to present exhaustively all features of LSX LC and LS:DO - there are many good books available with detail information on the subject. My purpose was to create a practical sample booklet which you could use as a reference when writing code.

The author that wrote this redpaper

Constantin Florea is an I/T Specialist from IBM Canada (since 1998); before 1998 he worked with IBM Romania (1991 - 1997). Extensive experience in software design, computer installation / administration (IBM PCs and RISC/6000), Lotus Notes, Web Servers installation/designing/administration. Constantin was part of the team that has managed installation/administration of BRIO OpEdge in IBM Canada, and Constantin solely developed an IBM Internal Security Tool (internal Web based) for supervising RISC/6000 boxes in IBM Canada. Accustomed with Lotus Notes environment -almost 8 years, when he was in charge with installation/administration of IBM Genie - Partner Info for IBM Romania. For the time being, he is working in Application Management Services (AMS) department - IBM Canada, developing code in Lotus Notes, other programming languages and administering boxes for IBM Shop PC Store application. He holds a degree in Software Development (Cybernetics) -1979, Academy for Economics Studies of Bucharest - Romania, Faculty of Economic Planning and Cybernetics.

This redpaper was compiled by Søren Peter Nielsen from ITSO Cambridge.

Lotus Notes Domino Connectivity.
A Collection of Examples

Constantin Florea
IBM CANADA

Internet: cflorea@ca.ibm.com
Notes Mail: Constantin Florea/Markham/IBM@IBMCA

Contents

1. Lotus Notes Domino Connectivity to Enterprise Data and Applications

In Lotus Notes Domino R5.x it is possible to maximize the power of Domino applications by using a range of features to get connected with Relational Databases(for example, Oracle or DB/2)

In respect of this, there are essentially two ways in which users can connect a Domino application to an external data source (RDBM's, file system, etc):

- LotusScript Extension for Lotus Domino Connectors (LSX LC)
- LotusScript Data Objects (LS:DO)

The purpose of this booklet is to demonstrate the use of these two techniques in accessing a DB/2 database and to give you a collection of samples encompassing the most important features. It was not my intent to present exhaustively all features of LSX LC and LS:DO - there are many good books available with detail information on the subject. My purpose was to create a practical sample booklet which you could use as a reference when writing code.

Throughout this booklet I make frequent references to the following books which you should have handy on your desk in order to get more details about the function used:

- Lotus Domino Release 5.0: A Developer's Handbook(IBM RedBook SG24-5331-01)
- Domino Release 5. Domino Enterprise Integration Guide.(It's part of Domino R5.x Documentation).
- Lotus Domino Designer R5. Domino Designer Programming Guide, Volume 2: LotusScript Classes(It's part of Domino R5.x Documentation).

In all above books there are full details about software structure of LSX LC, LS:DO, their strength and weaknesses and a plenty of advices about when and where it is recommended to use one or the other.

1.1 LSX LC

LSX LC provides access to a wide variety of external data sources through the following Connectors:

- DB2/UDB
- EDA/SQL
- File System
- Notes
- ODBC
- Oracle
- Sybase

For this purpose, LSX LC defines a set of classes for native access to those sources:
- LC_Session to handle available connectors and errors.
- LC_Connection to handle the connection to the data source.
- LC_FieldList to handle arrays of row data from the data source.
- LC_Field to handle individual data fields from the data source.
- LC_Currency, LC_Datetime, LC_Numeric, LC_Stream to handle individual data types.

LSX LC may be used alone or in conjunction with Domino Enterprise Connection Services(DECS).

Actually, as DECS was built using the set of LSX LC classes it should be seen as a real life application, built by Notes in order to allow the user an easy access to DBMS products. Therefore, the users can do a reverse engineering on DECS application, and build their own application.

In reality, DECS doesn't use all features of LSX LC, and frequently, the users prefer to build their own application based on LSX LC.

1.2 LS:DO

LS:DO is a LotusScript extension library that provides classes for working with Open DataBase Connectivity(ODBC).
For the time being, LS:DO supports ODBC Version 2.0 standard on a lot of platforms: Windows, OS/2, AIX, Solaris HP-UX.
LS:DO is a set of three LotusScript classes as follows:

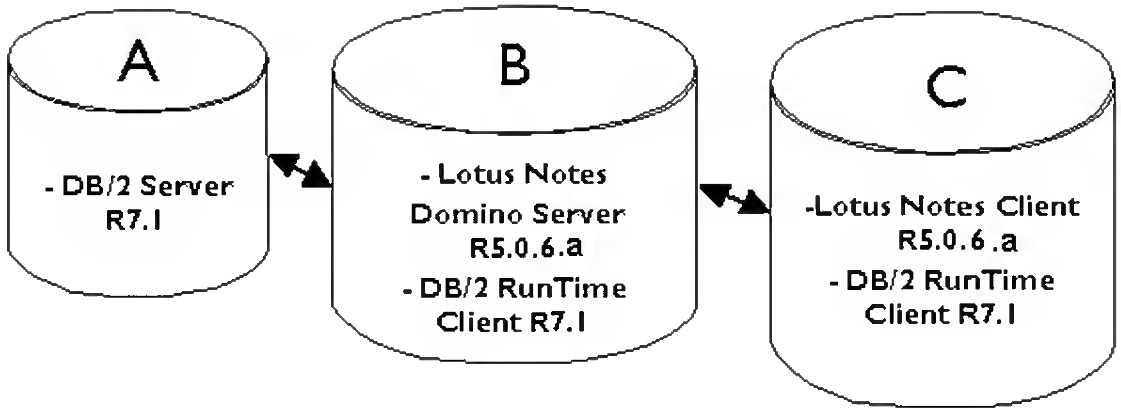
- ODBCConnection represents ODBC data access features for connecting to a data source.
- ODBCQuery represents the OBC data access features for defining an SQL statement.
- ODBCResultSet represents the ODBC data access features for performing operations on a result set.

* * *

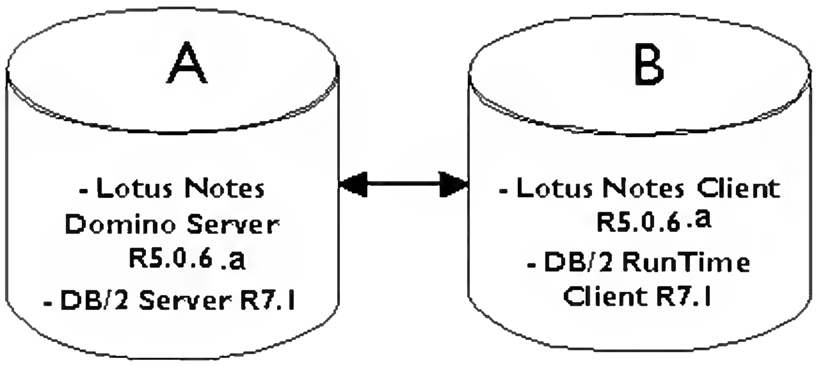
All examples in this booklet have been done using two configurations on the following hardware/software platform:
Intel Pentium III, Windows NT Workstation 4.00.1381, Token Ring Connection, TCP/IP Protocol.(see **Configuration I, Configuration II**)
Regarding DB/2, the examples use, SAMPLE database that was generated during the installation of DB/2 Server R7.1

In Chapter 3 of **Domino Release 5. Domino Enterprise Integration Guide** book is a very detailed description of prerequisites for DB/2 connection with Lotus Notes Domino.

Configuration I



Configuration II



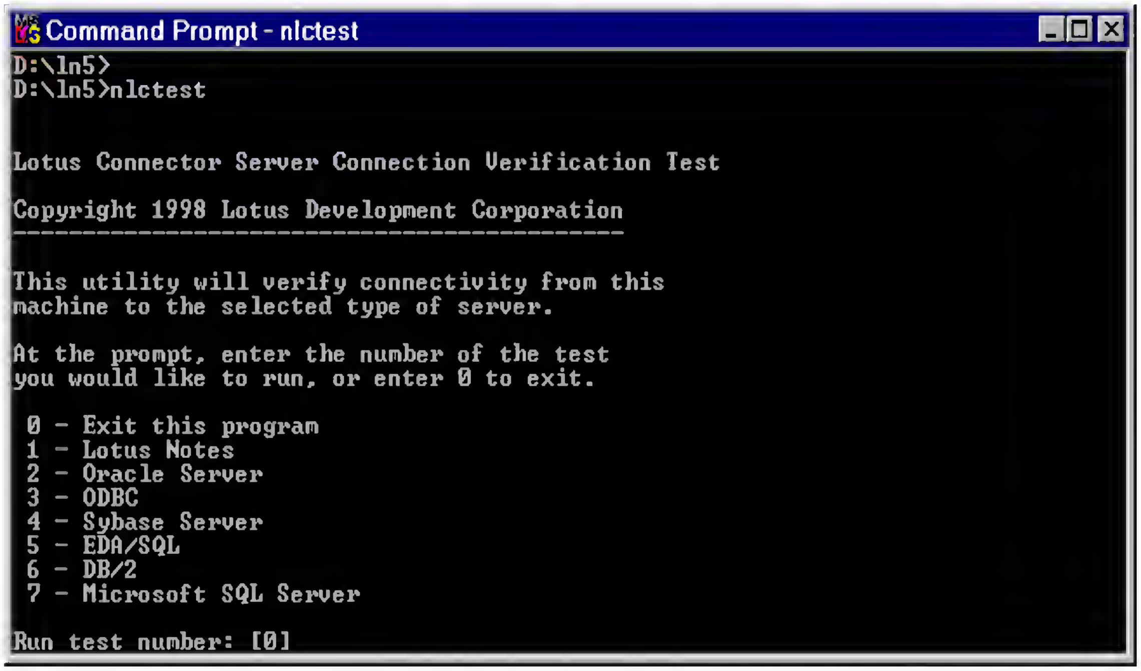
In **Configuration I**, a DB/2 RunTime Client R7.1 has been installed on B box for connecting with DB/2 Server R7.1 (which resides on Box A). That is because Lotus Notes Domino Server R5.x pushes/pulls information to/from box A, through this DB/2 Client.
If all Lotus Notes agents/codes run on Lotus Notes Domino Server only, the Lotus Notes Client (on box C) has no direct involvement in triggering manually any agents/codes containing LSCX or LS:DO, so it is no reason to have a DB/2 Client on box C.

In **Configuration II**, a DB/2 RunTime Client R7.1 is not required on box A, since DB/2 Server R7.1 takes care for connecting with Lotus Notes Domino Server R5.x.
If all Lotus Notes agents/codes run on Lotus Notes Domino Server only, the Lotus Notes Client (on box B) has no direct involvement in triggering manually any agents/codes containing LSCX or LS:DO, so it is no reason to have a DB/2 Client on box B.

In both configurations, before starting the examples, it's a good idea to check the connectivity to external data sources. Lotus Notes (Server and Client) comes with a test program named as follows:

- NLCTEST.EXE - for Windows 95/NT(Win32)
- ILCTEST.EXE - for OS/2
- ALCTEST.EXE for Windows NT/Alpha

When you run NLCTEST.EXE(in a DOS Box) the following screen brings-up:



```
D:\ln5>
D:\ln5>nlctest


Lotus Connector Server Connection Verification Test

Copyright 1998 Lotus Development Corporation
---------------------------------------------

This utility will verify connectivity from this
machine to the selected type of server.

At the prompt, enter the number of the test
you would like to run, or enter 0 to exit.

 0 - Exit this program
 1 - Lotus Notes
 2 - Oracle Server
 3 - ODBC
 4 - Sybase Server
 5 - EDA/SQL
 6 - DB/2
 7 - Microsoft SQL Server

Run test number: [0]
```

For **Configuration I** the following tests are required:

1. Locate and run NLCTEST.EXE on box B.
2. Select Option 3 for testing ODBC connection from box B to box A
3. Select Option 6 for testing DB/2 connection from box B to box A.

If you have installed DB/2 RunTime Client R7.1 on box C make the following tests:

1. Locate and run NLCTEST.EXE on box C.
2. Select Option 3 for testing ODBC connection from box C to box A
3. Select Option 6 for testing DB/2 connection from box C to box A.
4. Select Option 1 for testing Lotus Notes connection from box C to box B.

For **Configuration II** the following tests are required:

1. Locate and run NLCTEST.EXE on box A.
2. Select Option 3 for testing ODBC connection from box A to box A

3. Select Option 6 for testing DB/2 connection from box A to box A.

If you have installed DB/2 RunTime Client R7.1 on box B make the following tests:

1. Locate and run NLCTEST.EXE on box B.
2. Select Option 3 for testing ODBC connection from box B to box A
3. Select Option 6 for testing DB/2 connection from box B to box A.
4. Select Option 1 for testing Lotus Notes connection from box B to box A

It is mandatory that all above tests involving NLCTEST.EXE must run successfully in order to exercise all the examples in this booklet.

*

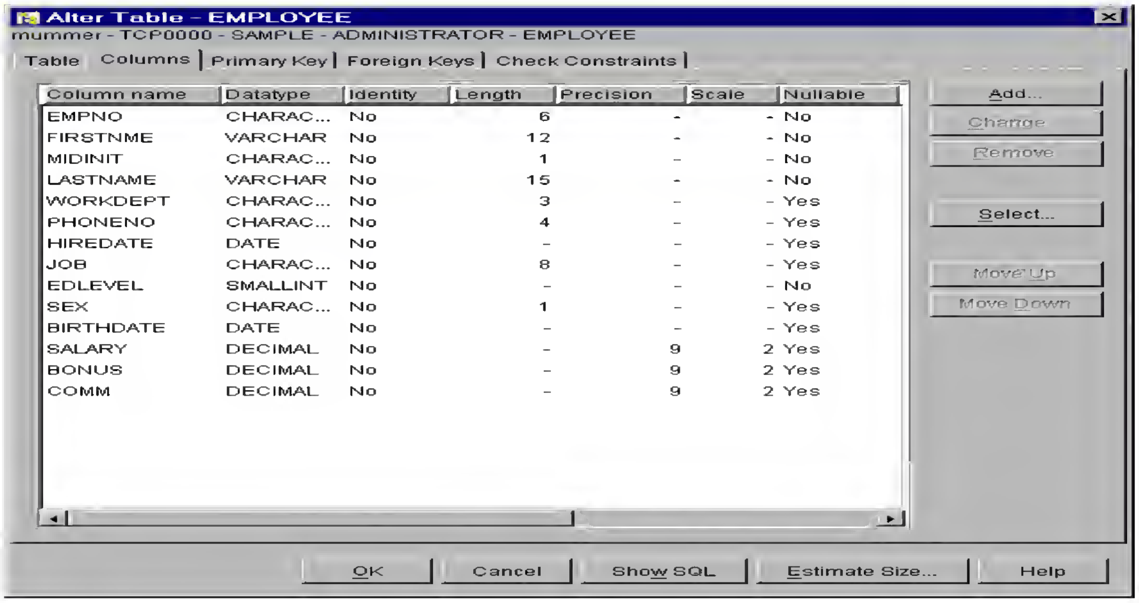
* *

As mentioned earlier, all the examples in this booklet work with SAMPLE DataBase, especially with two tables of it: EMPLOYEE and DEPARTMENT. In some examples we change the content of EMPLOYEE table and in some we create a new table(EUROPE - part of SAMPLE DataBase), populate it, print it, delete it.

The initial content of EMPLOYEE table is as follows:

```
EMPNO  FIRSTNME  MIDINIT LASTNAME  WORKDEPT PHONENO HIREDATE   JOB      EDLEVEL SEX BIRTHDATE  SALARY   BONUS    COMM
------ --------- ------- --------- -------- ------- ---------- -------- ------- --- ---------- ---------------- -------
000010 CHRISTINE    I    HAAS      A00      3978    01/01/1965 PRES     18        F 08/24/1933 52750.00 1000.00 4220.00
000020 MICHAEL      L    THOMPSON  B01      3476    10/10/1973 MANAGER  18        M 02/02/1948 41250.00  800.00 3300.00
000030 SALLY        A    KWAN      C01      4738    04/05/1975 MANAGER  20        F 05/11/1941 38250.00  800.00 3060.00
000050 JOHN         B    GEYER     E01      6789    08/17/1949 MANAGER  16        M 09/15/1925 40175.00  800.00 3214.00
000060 IRVING       F    STERN     D11      6423    09/14/1973 MANAGER  16        M 07/07/1945 32250.00  500.00 2580.00
000070 EVA          D    PULASKI   D21      7831    09/30/1980 MANAGER  16        F 05/26/1953 36170.00  700.00 2893.00
000090 EILEEN       W    HENDERSON E11      5498    08/15/1970 MANAGER  16        F 05/15/1941 29750.00  600.00 2380.00
000100 THEODORE     Q    SPENSER   E21      0972    06/19/1980 MANAGER  14        M 12/18/1956 26150.00  500.00 2092.00
000110 VINCENZO     G    LUCCHESSI A00      3490    05/16/1958 SALESREP 19        M 11/05/1929 46500.00  900.00 3720.00
000120 SEAN              O'CONNELL A00      2167    12/05/1963 CLERK    14        M 10/18/1942 29250.00  600.00 2340.00
000130 DOLORES      M    QUINTANA  C01      4578    07/28/1971 ANALYST  16        F 09/15/1925 23800.00  500.00 1904.00
000140 HEATHER      A    NICHOLLS  C01      1793    12/15/1976 ANALYST  18        F 01/19/1946 28420.00  600.00 2274.00
000150 BRUCE             ADAMSON   D11      4510    02/12/1972 DESIGNER 16        M 05/17/1947 25280.00  500.00 2022.00
000160 ELIZABETH    R    PIANKA    D11      3782    10/11/1977 DESIGNER 17        F 04/12/1955 22250.00  400.00 1780.00
000170 MASATOSHI    J    YOSHIMURA D11      2890    09/15/1978 DESIGNER 16        M 01/05/1951 24680.00  500.00 1974.00
000180 MARILYN      S    SCOUTTEN  D11      1682    07/07/1973 DESIGNER 17        F 02/21/1949 21340.00  500.00 1707.00
000190 JAMES        H    WALKER    D11      2986    07/26/1974 DESIGNER 16        M 06/25/1952 20450.00  400.00 1636.00
000200 DAVID             BROWN     D11      4501    03/03/1966 DESIGNER 16        M 05/29/1941 27740.00  600.00 2217.00
000210 WILLIAM      T    JONES     D11      0942    04/11/1979 DESIGNER 17        M 02/23/1953 18270.00  400.00 1462.00
000220 JENNIFER     K    LUTZ      D11      0672    08/29/1968 DESIGNER 18        F 03/19/1948 29840.00  600.00 2387.00
000230 JAMES        J    JEFFERSON D21      2094    11/21/1966 CLERK    14        M 05/30/1935 22180.00  400.00 1774.00
000240 SALVATORE    M    MARINO    D21      3780    12/05/1979 CLERK    17        M 03/31/1954 28760.00  600.00 2301.00
000250 DANIEL       S    SMITH     D21      0961    10/30/1969 CLERK    15        M 11/12/1939 19180.00  400.00 1534.00
000260 SYBIL        P    JOHNSON   D21      8953    09/11/1975 CLERK    16        F 10/05/1936 17250.00  300.00 1380.00
000270 MARIA        L    PEREZ     D21      9001    09/30/1980 CLERK    15        F 05/26/1953 27380.00  500.00 2190.00
000280 ETHEL        R    SCHNEIDER E11      8997    03/24/1967 OPERATOR 17        F 03/28/1936 26250.00  500.00 2100.00
000290 JOHN         R    PARKER    E11      4502    05/30/1980 OPERATOR 12        M 07/09/1946 15340.00  300.00 1227.00
000300 PHILIP       X    SMITH     E11      2095    06/19/1972 OPERATOR 14        M 10/27/1936 17750.00  400.00 1420.00
000310 MAUDE        F    SETRIGHT  E11      3332    09/12/1964 OPERATOR 12        F 04/21/1931 15900.00  300.00 1272.00
000320 RAMLAL       V    MEHTA     E21      9990    07/07/1965 FIELDREP 16        M 08/11/1932 19950.00  400.00 1596.00
000330 WING              LEE       E21      2103    02/23/1976 FIELDREP 14        M 07/18/1941 25370.00  500.00 2030.00
000340 JASON        R    GOUNOT    E21      5698    05/05/1947 FIELDREP 16        M 05/17/1926 23840.00  500.00 1907.00

  32 record(s) selected.
```

The initial structure of EMPLOYEE table is as follows:


Alter Table - EMPLOYEE

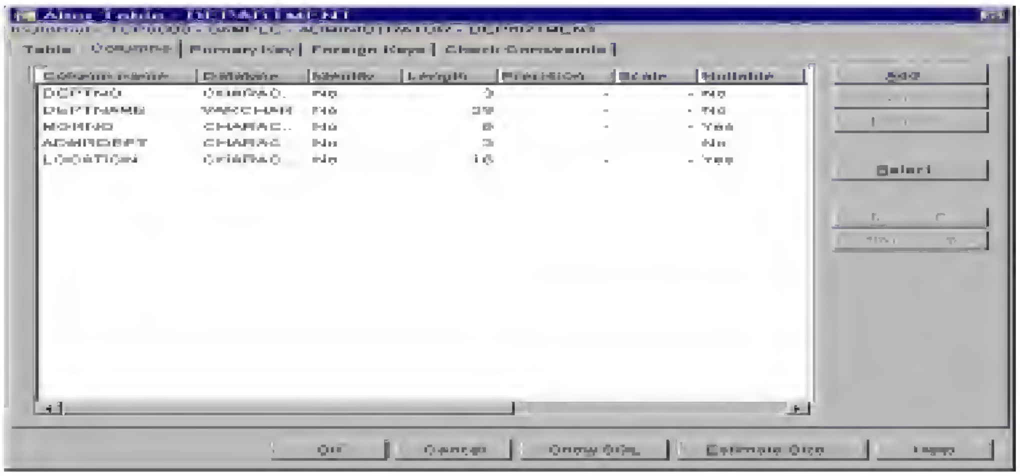
mummer - TCP0000 - SAMPLE - ADMINISTRATOR - EMPLOYEE

Column name	Datatype	Identity	Length	Precision	Scale	Nullable
EMPNO	CHARAC...	No	6	-	-	No
FIRSTNME	VARCHAR	No	12	-	-	No
MIDINIT	CHARAC...	No	1	-	-	No
LASTNAME	VARCHAR	No	15	-	-	No
WORKDEPT	CHARAC...	No	3	-	-	Yes
PHONENO	CHARAC...	No	4	-	-	Yes
HIREDATE	DATE	No	-	-	-	Yes
JOB	CHARAC...	No	8	-	-	Yes
EDLEVEL	SMALLINT	No	-	-	-	No
SEX	CHARAC...	No	1	-	-	Yes
BIRTHDATE	DATE	No	-	-	-	Yes
SALARY	DECIMAL	No	-	9	2	Yes
BONUS	DECIMAL	No	-	9	2	Yes
COMM	DECIMAL	No	-	9	2	Yes

The initial content of DEPARTMENT table is as follows:

```
DEPTNO DEPTNAME                      MGRNO  ADMRDEPT LOCATION
------ ----------------------------- ------ -------- --------
A00    SPIFFY COMPUTER SERVICE DIV.  000010 A00      -
B01    PLANNING                      000020 A00      -
C01    INFORMATION CENTER            000030 A00      -
D01    DEVELOPMENT CENTER            -      A00      -
D11    MANUFACTURING SYSTEMS         000060 D01      -
D21    ADMINISTRATION SYSTEMS        000070 D01      -
E01    SUPPORT SERVICES              000050 A00      -
E11    OPERATIONS                    000090 E01      -
E21    SOFTWARE SUPPORT              000100 E01      -

  9 record(s) selected.
```

The initial structure of DEPARTMENT table is as follows:


Alter Table - DEPARTMENT

mummer - TCP0000 - SAMPLE - ADMINISTRATOR - DEPARTMENT

Column name	Datatype	Identity	Length	Precision	Scale	Nullable
DEPTNO	CHARAC...	No	3	-	-	No
DEPTNAME	VARCHAR	No	29	-	-	No
MGRNO	CHARAC...	No	6	-	-	Yes
ADMRDEPT	CHARAC...	No	3			No
LOCATION	CHARAC...	No	16	-	-	Yes

Here is a short description of examples:

2. LotusScript Extension for Lotus Domino Connectors(LSX LC)

Example 2.1

This example displays the employees' LASTNAME and EMPNO from EMPLOYEE table of DB/2 SAMPLE database in a Notes document using EXECUTE method of LC_Connection class.

Example 2.2

This example displays information about a particular employee. The information is gathered from the tables EMPLOYEE and DEPARTMENT using EXECUTE method of LC_Connection class.

Example 2.3

This example displays all the rows from EMPLOYEE table using **"Nothing"** clause in SELECT method of LC_Connection class and FIELDNAMES property of LC_Connection. With the help of FIELDNAMES, there is the possibility of building a result set, based only on those fields of external database which we need; in this example we need to fetch fields EMPNO, LASTNAME, HIREDATE only.

Example 2.4

This example displays all the rows from EMPLOYEE table which contain the text **"JAMES"** in the field FIRSTNME using FIELDNAMES property of LC_Connection. With the help of FIELDNAMES, there is the possibility to build a result set, based only on those fields of external database which we need; in this example we need to fetch fields EMPNO, LASTNAME, HIREDATE only. In the present example, the text **JAMES** is hard coded, but you can build a construction, that asks you to type a name. As you can see, many opportunities exist for additional examples here.

Example 2.5

This example produces the same result as **EXAMPLE 2.4** following the same procedure but instead to use FIELDNAMES property of LC_Connection class, it makes use of LOOKUP method of LC_Fieldlist class.

Example 2.6

This example produces the same result as **EXAMPLE 2.4** following the same procedure but instead to use FIELDNAMES property of LC_Connection class, it makes use of CATALOG method of LC_Connection class.

Example 2.7

This example produces the same result as **EXAMPLE 2.4** following the same procedure but instead to use FIELDNAMES property of LC_Connection class, it makes use of MAP method of LC_Connection class.

Example 2.8

This example updates a row in EMPLOYEE table for an EMPNO value. It works with the document created in Example 2.1.

Example 2.9

This example creates a new table named EUROPE in SAMPLE database. The table will be empty, having the following structure:
CITY, text, 10 chars in size.
COUNTRY, text, 10 chars in size.
Following examples will show how to populate, update and delete records in this table.

Example 2.10

This example adds rows into the table created during the **EXAMPLE 2.9**, populating the field CITY with PARIS, and COUNTRY with FRANCE. In the present example, the texts **PARIS** and **FRANCE** are hard coded, but you can build a construction, that asks you to type a specific CITY and COUNTRY respectively. As you can see, many opportunities exist for additional examples here.

Example 2.11

This example deletes all rows into the table, created during the **EXAMPLE 2.9**, for which the column COUNTRY is **FRANCE.** In the present example, the text **FRANCE** is hard coded, but you can build a construction, that asks you to type a specific COUNTRY. As you can see, many opportunities exist for additional examples here.

Example 2.12

This example removes, using the method DROP of LC_Connection class, the table created during the **EXAMPLE 2.9**.

Example 2.13

This example retrieves a copy of the current value for a connection property. Actually it shows the values behind Property Token from **Appendix B** of **Domino Release 5. Domino Enterprise Integration Guide** book.

Example 2.14

This example retrieves all properties supported by a connector. Actually it shows the values behind Property Token from **Appendix B** and **Appendix C** of **Domino Release 5. Domino Enterprise Integration Guide** book.

Example 2.15

This example produces the same result as **Example 2.14** but brings -up more details about all properties supported by a connector.

Example 2.16

This example passes through all valid connectors of a Lotus Extension for Lotus Connectors installation. It gives you information from a Lotus Connector about its supported functionality and naming used by the backend systems as well as the sort of Flags supported by LC_Stream class

Example 2.17

This example passes through all valid MetaConnectors of a Lotus Extension for Lotus Connectors installation. It gives you information from a Lotus Connector about its supported functionality and naming used by the backend systems as well as the sort of Flags supported by LC_Stream class

Example 2.18

This example looks up a Connector name, gives all its features as well as the sort of Flags supported by LC_Stream class

Example 2.19

This example looks up a MetaConnector name, gives all its features as well as the sort of Flags supported by LC_Stream class

Example 2.20

This example shows the result of execution for a lot of methods, properties, passing through all LSX LC classes. To understand it, you should have aside, the print out of the example and to follow the code lines.

Example 2.21

This example shows how to access external databases via a Web browser and Domino Server, using LSX LC. To access the data from the Web browser, you must define a LSX LC connection to external data source and must write the LSX LC code in an agent that runs via a URL command. The display of the data needed to be formatted in HTML. In this example, giving the employee's serial number, we get information about an employee from SAMPLE database. **Example 2.21 is similar with Example 3.14; the only difference is that Example 2.21 uses LSX LC and Example 3.14 uses ODBC.**

3. LotusScript Data Object(LS:DO)

Example 3.1

This example displays the name of the available data sources

Example 3.2

This example shows an agent connection to the data source. If the connection fails the agent exits, contrary the agent lists the tables for the data source, looping through a string array returned by ListTables.

Example 3.3

This example passes through all rows of EMPLOYEE table and gets FIRSTNME and LASTNAME found in each row.

Example 3.4

This example sets the parameters in an SQL query then using NumParameters as upper bound, makes a loop in order to retrieve the row containing FIRSTNME and LASTNAME.

Example 3.5

This example examines all the fields (columns) in the EMPLOYEE table and displays their features

Example 3.6

This example shows an agent (AGENT6) that accesses all the rows of a result set twice, starting from the first row. The first time you do not explicitly set FirstRow since the first NextRow following an EXECUTE implicitly sets FirstRow. The second time, you must explicitly set FirstRow and process the first row before entering the loop.

Example 3.7

This example locates all the rows in a result set with **"JAMES"** in **"FIRSTNME"** field and **"DESIGNER** in field **2**.

Example 3.8

This example displays all rows in EMPLOYEE table, for each row showing the values of EMPNO, FIRSTNME, LASTNAME.The variable into which the result set value is stored, is also used as the second argument to GetValue in order to make the data typing explicitly.

Example 3.9

This example displays, just for the first row of EMPLOYEE table, the name of column, the type of column and the value of column.

Example 3.10

This example is based on a form and view, both named "PhoneBook. The form has three fields: lastName, firstName, phoneNumber. The view has seven Actions. The example also uses the agent AGENT11.
The following items are exercised:

- ACTION1: creates new table onto DB2 (named Phone), deletes a table (named Phone) adds new rows into the Phone table.
- ACTION2: adds new rows into the Phone table.
- ACTION3: deletes a row in the Phone table but if the row is unique only; that means there aren't two columns in the Phone table having the same LASTNAME, FIRSTNAME.
- ACTION4: displays all rows of the Phone table using the sequence:

```
DO
    RESULT.NEXTROW
.
.
LOOP UNTIL RESULT.ISENDOFDATA
```

- ACTION5: DROPs the table Phone
- ACTION6: updates the column FIRSTNAME for the row FLOREA COSTICA 123456, changing COSTICA with CRISTINA
- ACTION7: displays all the rows of the Phone table using the sequence:

```
RESULT.LASTROW
FOR I=1 to RESULT.NUMROWS
   .
   .
NEXT
```

- AGENT11: deletes all rows from the Phone table, emptying the Phone table, but does not remove the Phone table. ACTION5 removes the Phone table.

Example 3.11

In this example there is the form FORM2 that contains two fields (text + editable) named dataSource and Table, four buttons named "Data Source", "Table", Postopen", "QueryClose", and two actions named "List Fields" and "List Procedure".

The button "Postopen" sets the objects, gets the names of the available data sources, writes the first one to the dataSource field, gets the names of the tables for the data source and writes the first one to the Table field.

The button “Data Source” writes the name of the next data source to the dataSource field, gets the tables for the new data source and writes the first one to the Table field.

The button “Table” writes the name of the next table to the Table field.

The action “List Fields” displays the names of all the fields for the current data source and table.

The action “List Procedures” displays the name of all the procedures for the current data source.

Example 3.12

In this example, each time when you exit from the field Part_Number (inside of which you must type a valid serial number taken from EMPNO of EMPLOYEE table), the code associated with this field, automatically fills in the fields Part_Name (with the value of FIRSTNME), Price (with the value of LASTNAME), Description (with the value of WORKDEPT).

Example 3.13

In order to understand this example, read the paragraph **“Tips and techniques - Handling an ODBC event**” from the book **Domino Release 5. Domino Designer Programming Guide, Volume 2.**

In this example, the values of a row in an ODBC table are displayed as fields on FORM4. The user can use buttons to get the next and previous rows. The event handler **AfterPositionChange** displays the number of the current row in another field on the form FORM4

Example 3.14

This example shows how to access external databases via a Web browser and Domino Server, using ODBC. To access the data from the Web browser, you must define an ODBC connection to external data source and must write the ODBC code in an agent that runs via a URL command. The display of the data needed to be formatted in HTML. In this example, giving the employee’s serial number, we get information about an employee from SAMPLE database. **Example 3.14 is similar with Example 2.21; the only difference is that Example 2.21 uses LSX LC and Example 3.14 uses ODBC.**

2. LotusScript Extension for Lotus Domino Connectors(LSX LC)

All examples in this chapter deal with LSX LC. To follow the exercises presented here, please create a Lotus Notes Database (our example LSXCODBC.NSF) from a blank template on Lotus Notes Domino Server(our example MUMMER.ISM.CAN.IBM.COM)

When you decide to study the examples of this Chapter, you should have aside the following books:

- Lotus Domino Release 5.0: A Developer's Handbook(IBM RedBook SG24-5331-01)
- Domino Release 5. Domino Enterprise Integration Guide.(It's part of Domino R5.x Documentation).

Example 2.1

This example displays the employees' LASTNAME and EMPNO from EMPLOYEE table of DB/2 SAMPLE database in a Notes document using EXECUTE method of LC_Connection class.
In order to achieve this objective do the following steps:

Step A - 2.1

Create a form on LSXCODBC.NSF, named FORM1 having the following structure:

- Cimp1: text + editable
- EmpNo: dialog list + editable, * allow multiple values, Control-> Choices: EmpNoList
- EmpNoList: text + editable
- FirstNme, MidInit, LastName, Sex, Bonus, Comm, Salary, PhoneNo, Job, WorkDept, EdLevel, DeptName, ManagerNo, Manager: text + editable
- BirthDate, HireDate: date/time + editable
- EmpNoAlias: text + computed, formula: EmpNo
- Name_Display: text + computed for display, formula: FirstNme+" "+MidInit+" "+LastName
- Sex_Display: text + computed for display, formula: Sex
- BirthDate_Display: date/time + computed for display, formula: BirthDate
- Bonus_Display: text + computed for display, formula: Bonus
- Comm_Display: text + computed for display, formula: Comm
- Salary_Display: text + computed for display, formula: Salary
- HireDate_Display: date/time + computed for display, formula: HireDate
- PhoneNo_Display: text + computed for display, formula: PhoneNo
- Job_Display: text + computed for display, formula: Job
- WorkDept_Display: text + computed for display, formula: WorkDept
- DeptName_Display: text + computed for display, formula: DeptName
- Manager_Display: text + computed for display, formula: Manager
- EdLevel_Display: text + computed for display, formula: EdLevel

Step B - 2.1

Select FORM1->Globals->Option Public
USELSX "*LSXLC"
The effect of USELSX "*LSXLC" is to invoke LotusScript Extensions for Connectors

Step C - 2.1

Create the following LotusScript code for BUTTON1:

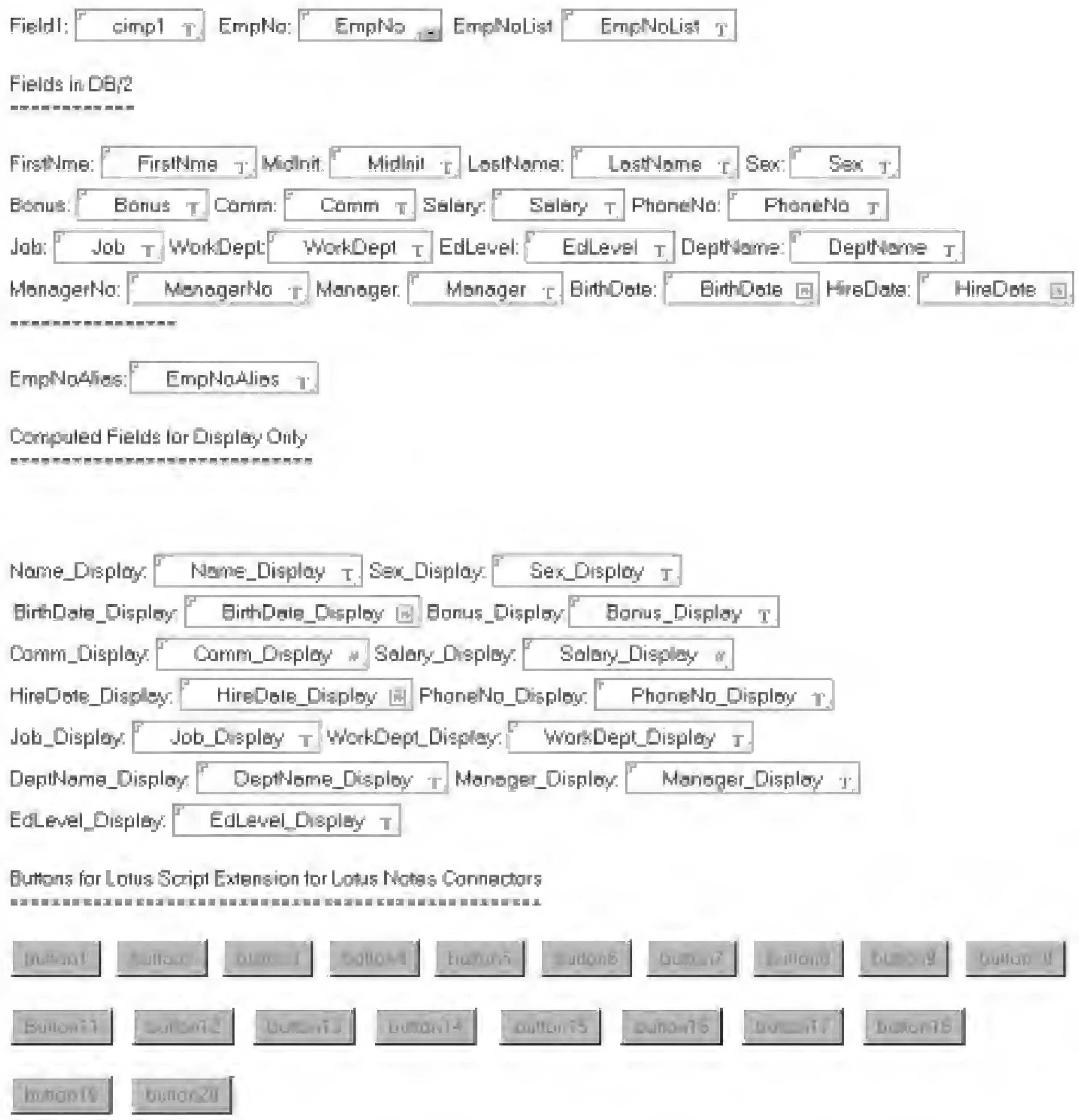
```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim LC_Conn As New LCConnection("db2")
        Dim LC_FldLst As New LCFieldList(1)
        Dim LC_Field1 As New LCField(LCTYPE_TEXT,1)
        Dim LC_Field2 As New LCField(LCTYPE_TEXT,1)
        Dim count As Long
        Dim SelectStatement As String
        Dim workspace As New notesuiworkspace
        Dim uidoc As notesuidocument
        Set uidoc=workspace.currentdocument
        On Error Goto ErrorHandler
        LC_Conn.Userid="Administrator"
        LC_Conn.Password="rac4you"
        LC_Conn.Database="SAMPLE"
        LC_Conn.Disconnect
        LC_S.ClearStatus
        LC_Conn.Connect
        SelectStatement="SELECT * FROM EMPLOYEE ORDER BY LASTNAME"
        count=LC_Conn.Execute(SelectStatement, LC_FldLst)
        If count <> 0 Then
                count=LC_Conn.Fetch(LC_FldLst,1,1)
                Set LC_Field1=LC_FldLst.GetField(1)
                Set LC_Field2=LC_FldLst.GetField(4)
                IDs=""
                Messagebox "The Loop is starting"
                While (count > 0) And LC_S.Status=LC_Success
                        IDs=IDs + LC_Field2.text(0) + "|" + LC_Field1.text(0) + ","
                        count=LC_Conn.Fetch(LC_FldLst,1,1)
                Wend
                Messagebox "The Loop is finished"
                Call uidoc.FieldSetText("EmpNoList", IDs)
                Call uidoc.refresh()
        End If
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
                status=LC_S.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

*

*   *

For the time being don't care about formulas behind the buttons: 2, 3.......20

The structure of FORM1 is as follows:



Field1: cimp1 EmpNo: EmpNo EmpNoList EmpNoList

Fields in DB/2
============

FirstNme: FirstNme MidInit MidInit LastName: LastName Sex: Sex

Bonus: Bonus Comm: Comm Salary: Salary PhoneNo: PhoneNo

Job: Job WorkDept WorkDept EdLevel: EdLevel DeptName: DeptName

ManagerNo: ManagerNo Manager: Manager BirthDate: BirthDate HireDate: HireDate
================

EmpNoAlias: EmpNoAlias

Computed Fields for Display Only
=============================

Name_Display: Name_Display Sex_Display: Sex_Display

BirthDate_Display: BirthDate_Display Bonus_Display: Bonus_Display

Comm_Display: Comm_Display Salary_Display: Salary_Display

HireDate_Display: HireDate_Display PhoneNo_Display: PhoneNo_Display

Job_Display: Job_Display WorkDept_Display: WorkDept_Display

DeptName_Display: DeptName_Display Manager_Display: Manager_Display

EdLevel_Display: EdLevel_Display

Buttons for Lotus Script Extension for Lotus Notes Connectors
===

In order to run **Example 2.1**, create a document using FORM1 and when the document is opened, write something in cimp1- let say alpha, and push onto BUTTON1. Your document will be populated and the document looks like below(save the document):

Field1: alpha EmpNo: EmpNoList: ADAMSON|000150; BROWN|000200; GEYER|000050; GOUNOT|000340; HAAS|000010; HENDERSON|000090; JEFFERSON|000230; JOHNSON|000260; JONES|000210; KWAN|000030; LEE|000330; LUCCHESSI|000110; LUTZ|000220; MARINO|000240; MEHTA|000320; NICHOLLS|000140; O'CONNELL|000120; PARKER|000290; PEREZ|000270; PIANKA|000160; PULASKI|000070; QUINTANA|000130; SCHNEIDER|000280; SCOUTTEN|000180; SETRIGHT|000310;

SMITH|000250; SMITH|000300; SPENSER|000100; STERN|000060; THOMPSON|000020; WALKER|000190; YOSHIMURA|000170

Fields in DB/2
============

FirstNme: MidInit: LastName: Sex:
Bonus: Comm: Salary: PhoneNo:
Job: WorkDept: EdLevel: DeptName:
ManagerNo: Manager: BirthDate: HireDate:
================

EmpNoAlias:

Computed Fields for Display Only
============================

Name_Display: Sex_Display:
BirthDate_Display: Bonus_Display:
Comm_Display: Salary_Display:
HireDate_Display: PhoneNo_Display:
Job_Display: WorkDept_Display:
DeptName_Display: Manager_Display:
EdLevel_Display:

Buttons for LotusScript Extension for Lotus Notes Connectors
==

*

* *

Let’s try to explain what happened.

Set up the connection using userid, password, and database name to get connected to:

```
Dim LC_Conn As New LCConnection("db2")
Dim LC_FldLst As New LCFieldList(1)
Dim LC_Field1 As New LCField(LCTYPE_TEXT,1)
Dim LC_Field2 As New LCField(LCTYPE_TEXT,1)
Dim count As Long
Dim SelectStatement As String
Dim workspace As New notesuiworkspace
Dim uidoc As notesuidocument
Set uidoc=workspace.currentdocument
On Error Goto ErrorHandler
LC_Conn.Userid="Administrator"
LC_Conn.Password="rac4you"
LC_Conn.Database="SAMPLE"
```

In order to clean up any previous aborted sessions, force a disconnection and reset it to normal.

```
LC_Conn.Disconnect
LC_S.ClearStatus
```

Do a connection.

```
LC_Conn.Connect
```

Create an SQL select command to retrieve the data for all columns from EMPLOYEE table, ordering them by LASTNAME column.

```
SelectStatement="SELECT * FROM EMPLOYEE ORDER BY LASTNAME"
```

Execute the SQL statement on the connection returning the values in to LC_FldLst variable. Get the number of rows returned in count variable. If you get count= -1, that means the number of rows is undetermined (that isn't an error).

```
count=LC_Conn.Execute(SelectStatement, LC_FldLst)
```

Step through each row returned from SQL Select statement. If the value returned from SQL statement is not zero, that means is no error, fetch a field list record from data source.

```
If count <> 0 Then
            count=LC_Conn.Fetch(LC_FldLst,1,1)
```

Put the values stored in columns 1(EMPNO) and 4(LASTNAME) of EMPLOYEE table into LC_Field1, LC_Field2.

```
Set LC_Field1=LC_FldLst.GetField(1)
            Set LC_Field2=LC_FldLst.GetField(4)
```

While the variable count is greater than zero(there are still rows to retrieve from the data source) and the Lotus Connectors session status is OK, get each record from data source:

```
IDs=""
            Messagebox "The Loop is starting"
            While (count > 0) And LC_S.Status=LC_Success
```

Set the value of the variable IDs using LASTNAME and EMPNO until all rows are read and store IDs value into EmpNoList field.

```
            IDs=IDs + LC_Field2.text(0) + "|" + LC_Field1.text(0) + ","
            count=LC_Conn.Fetch(LC_FldLst,1,1)
      Wend
      Messagebox "The Loop is finished"
      Call uidoc.FieldSetText("EmpNoList", IDs)
```

When everything is finished, the field EmpNoList contains a string like LASTNAME | EMPNO and EmpNo field contains a dialog list of type LASTNAME. Actually behind each LASTNAME visualized in EmpNo field there is an alias composed of EMPNO value.

Example 2.2

This example displays information about a particular employee. The information is gathered from the tables EMPLOYEE and DEPARTMENT using EXECUTE method of LC_Connection class.

In order to achieve this objective, do the following step:

Step A - 2.2

Create the following LotusScript code for BUTTON2:

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim LC_Conn As New LCConnection("db2")
        Dim LC_FldLst As New LCFieldList(1)
        Dim LC_FldLst2 As New LCFieldList(1)
        Dim LC_Field As New LCField(LCTYPE_TEXT,1)
        Dim count As Long
        Dim SelectStatement As String
        Dim workspace As New notesuiworkspace
        Dim uidoc As notesuidocument
        Set uidoc=workspace.currentdocument
        On Error Goto ErrorHandler
        LC_Conn.Userid="Administrator"
        LC_Conn.Password="rac4you"
        LC_Conn.Database="SAMPLE"
        LC_Conn.Disconnect
        LC_S.ClearStatus
        LC_Conn.Connect
        EmpNo=uidoc.fieldgettext("EmpNoAlias")
        SelectStatement="SELECT * FROM EMPLOYEE WHERE EMPNO = '" & EmpNo & "'"
        count=LC_conn.Execute(SelectStatement, LC_FldLst)
        If count <> 0 Then
                count=LC_Conn.Fetch(LC_FldLst,1,1)
                Call uidoc.fieldsettext("FirstNme",LC_FldLst.FIRSTNME(0))
                Call uidoc.fieldsettext("LastName",LC_FldLst.LASTNAME(0))
                Call uidoc.fieldsettext("MIdInit",LC_FldLst.MIDINIT(0))
                Call uidoc.fieldsettext("Sex",LC_FldLst.SEX(0))
                Set dt_TempDate=New Notesdatetime(LC_FldLst.BIRTHDATE(0))
                Call uidoc.fieldsettext("BirthDate",dt_TempDate.DateOnly)
                Call uidoc.fieldsettext("Bonus",Cstr(LC_FldLst.BONUS(0)))
                Call uidoc.fieldsettext("Comm",Cstr(LC_FldLst.COMM(0)))
                Call uidoc.fieldsettext("Salary",Cstr(LC_FldLst.SALARY(0)))
                Set dt_TempDate=New Notesdatetime(LC_FldLst.HIREDATE(0))
                Call uidoc.fieldsettext("HireDate",dt_TempDate.DateOnly)
                Call uidoc.fieldsettext("PhoneNo",LC_FldLst.PHONENO(0))
                Call uidoc.fieldsettext("Job",LC_FldLst.JOB(0))
                Call uidoc.fieldsettext("WorkDept",LC_FldLst.WORKDEPT(0))
                Call uidoc.fieldsettext("EdLevel",Cstr(LC_FldLst.EDLEVEL(0)))
                If LC_FldLst.WORKDEPT(0) <> "" Then
                        SelectStatement="SELECT * FROM DEPARTMENT WHERE DEPTNO = '" &
LC_FldLst.WORKDEPT(0) & "'"
                        count=LC_Conn.Execute(SelectStatement, LC_FldLst2)
                        If count <> 0 Then
```

```
                                        count=LC_Conn.Fetch(LC_FldLst2,1,1)
                                        Call uidoc.fieldsettext("DeptName",LC_FldLst2.DEPTNAME(0))
                                        Call uidoc.fieldsettext("ManagerNo",LC_FldLst2.MGRNO(0))
                                        SelectStatement="SELECT * FROM EMPLOYEE WHERE EMPNO = '" &
LC_FldLst2.MGRNO(0) & "'"
                                        Set LC_FldLst=New LCFieldList(1)
                                        count=LC_conn.Execute(SelectStatement, LC_FldLst)
                                        If count <> 0 Then
                                                count=LC_Conn.Fetch(LC_FldLst,1,1)
                                                Call uidoc.fieldsettext("Manager",LC_FldLst.LastName(0))
                                        End If
                                End If
                        End If
                        Call uidoc.refresh
                End If
                End
ErrorHandler:
                Messagebox "Attention ! You are in Error"
                Dim msg As String
                Dim errortext As String
                Dim msgcode As Long
                Dim status As Integer
                If (LC_S.status <> LCSUCCESS) Then
                        status=LC_S.getstatus(errortext,msgcode,msg)
                        Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                        & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
                Else
                        Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
                End If
                End
End Sub
```

In order to run **Example 2.2,** do the following steps:

- ✓ Open, in edit mode, the document created in Example 2.1
- ✓ Select a name(LASTNAME) from the field EmpNo - let say ADAMSON. The field EmpNoAlias, automatically will contain the EMPNO value (000150) for this LASTNAME.
- ✓ Push onto BUTTON2. Your document will be populated in areas **Fields in DB/2** and **Computed Fields for Display Only** and the document looks like below(save the document):

```
Field1: alpha  EmpNo: ADAMSON  EmpNoList: ADAMSON|000150; BROWN|000200; GEYER|000050;
GOUNOT|000340; HAAS|000010; HENDERSON|000090; JEFFERSON|000230; JOHNSON|000260;
JONES|000210; KWAN|000030; LEE|000330; LUCCHESSI|000110; LUTZ|000220; MARINO|000240;
MEHTA|000320; NICHOLLS|000140; O'CONNELL|000120; PARKER|000290; PEREZ|000270;
PIANKA|000160; PULASKI|000070; QUINTANA|000130; SCHNEIDER|000280; SCOUTTEN|000180;
SETRIGHT|000310; SMITH|000250; SMITH|000300; SPENSER|000100; STERN|000060; THOMPSON|000020;
WALKER|000190; YOSHIMURA|000170

Fields in DB/2
============

FirstNme: BRUCE MidInit:  LastName: ADAMSON Sex: M
Bonus: 500 Comm: 2022 Salary: 25280 PhoneNo: 4510
Job: DESIGNER WorkDept:D11 EdLevel: 16 DeptName: MANUFACTURING SYSTEMS
ManagerNo: 000060 Manager: STERN BirthDate: 05/17/1947 HireDate: 02/12/72
================
```

EmpNoAlias:000150

Computed Fields for Display Only
==============================

Name_Display: BRUCE ADAMSON Sex_Display: M
 BirthDate_Display: 05/17/1947 Bonus_Display:500
Comm_Display: 2022 Salary_Display: 25280
HireDate_Display: 02/12/72 PhoneNo_Display: 4510
Job_Display: DESIGNER WorkDept_Display: D11
DeptName_Display: MANUFACTURING SYSTEMS Manager_Display: STERN
EdLevel_Display: 16

Buttons for LotusScript Extension for Lotus Notes Connectors
==

*

*　　　　*

Let’s try to explain what happened.

The value stored in the EmpNoAlias field (the alias - that means EMPNO value not LASTNAME value) is retrieved into EmpNo variable.

```
  EmpNo=uidoc.fieldgettext("EmpNoAlias")
          SelectStatement="SELECT * FROM EMPLOYEE WHERE EMPNO = '" & EmpNo & "'"
```

Check to see if the value of count is zero; if not, fetch the first row from result set.

```
  If count <> 0 Then
          count=LC_Conn.Fetch(LC_FldLst,1,1)
```

Retrieve the rows from result set and put them into the fields of form.

```
          Call uidoc.fieldsettext("FirstNme",LC_FldLst.FIRSTNME(0))
          Call uidoc.fieldsettext("LastName",LC_FldLst.LASTNAME(0))
          Call uidoc.fieldsettext("MIdInit",LC_FldLst.MIDINIT(0))
          Call uidoc.fieldsettext("Sex",LC_FldLst.SEX(0))
          Set dt_TempDate=New Notesdatetime(LC_FldLst.BIRTHDATE(0))
          Call uidoc.fieldsettext("BirthDate",dt_TempDate.DateOnly)
          Call uidoc.fieldsettext("Bonus",Cstr(LC_FldLst.BONUS(0)))
          Call uidoc.fieldsettext("Comm",Cstr(LC_FldLst.COMM(0)))
          Call uidoc.fieldsettext("Salary",Cstr(LC_FldLst.SALARY(0)))
          Set dt_TempDate=New Notesdatetime(LC_FldLst.HIREDATE(0))
```

```
                    Call uidoc.fieldsettext("HireDate",dt_TempDate.DateOnly)
                    Call uidoc.fieldsettext("PhoneNo",LC_FldLst.PHONENO(0))
                    Call uidoc.fieldsettext("Job",LC_FldLst.JOB(0))
                    Call uidoc.fieldsettext("WorkDept",LC_FldLst.WORKDEPT(0))
                    Call uidoc.fieldsettext("EdLevel",Cstr(LC_FldLst.EDLEVEL(0)))
```

To retrieve the department name, the DEPARTMENT table is querried using the value retrieved from WORKDEPT field in the EMPLOYEE table.

```
          If LC_FldLst.WORKDEPT(0) <> "" Then
                    SelectStatement="SELECT * FROM DEPARTMENT WHERE DEPTNO = '" & 
LC_FldLst.WORKDEPT(0) & "'"
```

In the end the manager's name is retrieved from EMPLOYEE table using the value retrieved from the MGRNO field from DEPARTMENT table.

```
                              SelectStatement="SELECT * FROM EMPLOYEE WHERE EMPNO = '" & 
LC_FldLst2.MGRNO(0) & "'"
```

Example 2.3

This example displays all the rows from EMPLOYEE table using **"Nothing"** clause in SELECT method of LC_Connection class and FIELDNAMES property of LC_Connection. With the help of FIELDNAMES, there is the possibility of building a result set, based only on those fields of external database which we need; in this example we need to fetch fields EMPNO, LASTNAME, HIREDATE only.
In order to achieve this objective do the following step:

Step A - 2.3

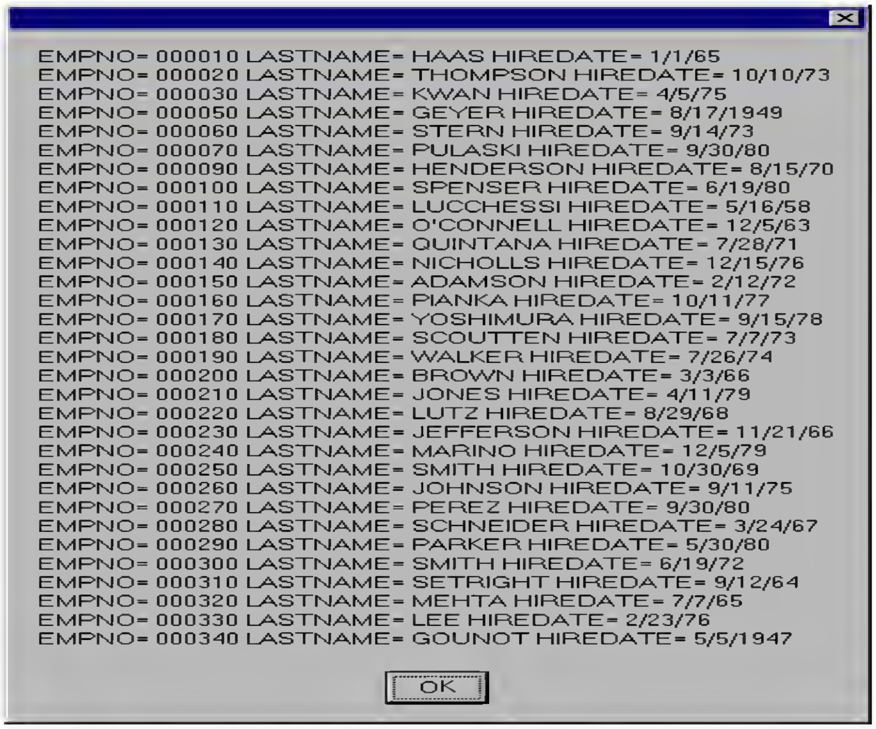
Create the following LotusScript code for BUTTON6:

```
Sub Click(Source As Button)
        On Error Goto handler
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        Dim session As New lcsession
        Dim src As New lcconnection("db2")
        Dim fields As New lcfieldlist
        session.clearstatus
        src.database="SAMPLE"
        src.userid="Administrator"
        src.password="rac4you"
        src.connect
        src.metadata="EMPLOYEE"
        If (src.select(Nothing,1,fields)=0) Then
                Messagebox "Error in Selection"
                End
        End If
        src.fieldnames="EMPNO,LASTNAME,HIREDATE"
        msg1=""
        While (src.fetch(fields)>0)
                msg1=msg1 & "EMPNO= " & fields.EMPNO(0) & " LASTNAME= " & fields.LASTNAME(0) _
                & " HIREDATE= " & fields.HIREDATE(0)  & Chr(10)
        Wend
        Messagebox msg1
        End
handler:
        If (session.status <> LCSUCCESS) Then
                status=session.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status &
Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " &
Err()
        End If
        End
End Sub
```

In order to run **Example 2.3** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON6.

The result is as follows:



```
EMPNO= 000010 LASTNAME= HAAS HIREDATE= 1/1/65
EMPNO= 000020 LASTNAME= THOMPSON HIREDATE= 10/10/73
EMPNO= 000030 LASTNAME= KWAN HIREDATE= 4/5/75
EMPNO= 000050 LASTNAME= GEYER HIREDATE= 8/17/1949
EMPNO= 000060 LASTNAME= STERN HIREDATE= 9/14/73
EMPNO= 000070 LASTNAME= PULASKI HIREDATE= 9/30/80
EMPNO= 000090 LASTNAME= HENDERSON HIREDATE= 8/15/70
EMPNO= 000100 LASTNAME= SPENSER HIREDATE= 6/19/80
EMPNO= 000110 LASTNAME= LUCCHESSI HIREDATE= 5/16/58
EMPNO= 000120 LASTNAME= O'CONNELL HIREDATE= 12/5/63
EMPNO= 000130 LASTNAME= QUINTANA HIREDATE= 7/28/71
EMPNO= 000140 LASTNAME= NICHOLLS HIREDATE= 12/15/76
EMPNO= 000150 LASTNAME= ADAMSON HIREDATE= 2/12/72
EMPNO= 000160 LASTNAME= PIANKA HIREDATE= 10/11/77
EMPNO= 000170 LASTNAME= YOSHIMURA HIREDATE= 9/15/78
EMPNO= 000180 LASTNAME= SCOUTTEN HIREDATE= 7/7/73
EMPNO= 000190 LASTNAME= WALKER HIREDATE= 7/26/74
EMPNO= 000200 LASTNAME= BROWN HIREDATE= 3/3/66
EMPNO= 000210 LASTNAME= JONES HIREDATE= 4/11/79
EMPNO= 000220 LASTNAME= LUTZ HIREDATE= 8/29/68
EMPNO= 000230 LASTNAME= JEFFERSON HIREDATE= 11/21/66
EMPNO= 000240 LASTNAME= MARINO HIREDATE= 12/5/79
EMPNO= 000250 LASTNAME= SMITH HIREDATE= 10/30/69
EMPNO= 000260 LASTNAME= JOHNSON HIREDATE= 9/11/75
EMPNO= 000270 LASTNAME= PEREZ HIREDATE= 9/30/80
EMPNO= 000280 LASTNAME= SCHNEIDER HIREDATE= 3/24/67
EMPNO= 000290 LASTNAME= PARKER HIREDATE= 5/30/80
EMPNO= 000300 LASTNAME= SMITH HIREDATE= 6/19/72
EMPNO= 000310 LASTNAME= SETRIGHT HIREDATE= 9/12/64
EMPNO= 000320 LASTNAME= MEHTA HIREDATE= 7/7/65
EMPNO= 000330 LASTNAME= LEE HIREDATE= 2/23/76
EMPNO= 000340 LASTNAME= GOUNOT HIREDATE= 5/5/1947
```

Example 2.4

This example displays all the rows from EMPLOYEE table which contain the text **“JAMES”** in the field FIRSTNME using FIELDNAMES property of LC_Connection.
With the help of FIELDNAMES, there is the possibility to build a result set, based only on those fields of external database which we need; in this example we need to fetch fields EMPNO, LASTNAME, HIREDATE only. In the present example, the text **JAMES** is hard coded, but you can build a construction, that asks you to type a name. As you can see, many opportunities exist for additional examples here.

Take care to the following remarked code in Step A - 2.4:

REM If you want to get all rows which don’t contain the key “JAMES”,
REM do OR with LCFIELDF_KEY_NE as in the first below line of code:
REM field.flags=LCFIELDF_KEY Or LCFIELDF_KEY_NE
REM It’s manadtory that the line field.flags=........... preceeds the line field.text=”JAMES”

In order to achieve the objective to fetch the rows do the following step:

Step A - 2.4

Create the following LotusScript code for BUTTON4:

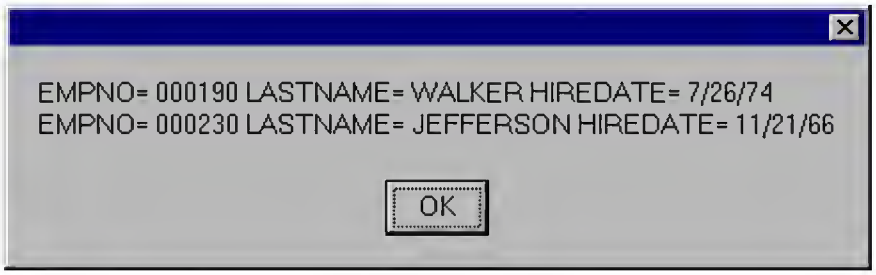
```
Sub Click(Source As Button)
        On Error Goto handler
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        Dim session As New lcsession
        Dim src As New lcconnection("db2")
        Dim keys As New lcfieldlist
        Dim fields As New lcfieldlist
        Dim field As lcfield
        session.clearstatus
        src.database="SAMPLE"
        src.userid="Administrator"
        src.password="rac4you"
        src.connect
        src.metadata="EMPLOYEE"
        Set field=keys.append("FIRSTNME",LCTYPE_TEXT)
        field.flags=LCFIELDF_KEY

REM If you want to get all rows which don’t contain the key “JAMES”,
REM do OR with LCFIELDF_KEY_NE as in the first below line of code:
REM field.flags=LCFIELDF_KEY Or LCFIELDF_KEY_NE
REM It’s mandatory that the line field.flags=........... preceeds the line field.text=”JAMES”

        field.text="JAMES"
        If (src.select(keys,1,fields)=0) Then
                Messagebox "Error in Selection"
                End
        End If
        src.fieldnames="EMPNO,LASTNAME,HIREDATE"
```

```
          msg1=""
          While (src.fetch(fields)>0)
                    msg1=msg1 & "EMPNO= " & fields.EMPNO(0) & " LASTNAME= " & fields.LASTNAME(0) _
                    & " HIREDATE= " & fields.HIREDATE(0) & Chr(10)
          Wend
          Messagebox msg1
          End
handler:
          If (session.status <> LCSUCCESS) Then
                    status=session.getstatus(errortext,msgcode,msg)
                    Messagebox "Internal Text Error= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                    & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
          Else
                    Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
          End If
          End
End Sub
```

In order to run **Example 2.4** do the following steps:

✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
✓ Push onto BUTTON4.

The result is as follows:



```
EMPNO= 000190 LASTNAME= WALKER HIREDATE= 7/26/74
EMPNO= 000230 LASTNAME= JEFFERSON HIREDATE= 11/21/66
```

Example 2.5

This example produces the same result as **EXAMPLE 2.4** following the same procedure but instead to use FIELDNAMES property of LC_Connection class, it makes use of LOOKUP method of LC_Fieldlist class.

In order to achieve this objective do the following step:

<u>Step A - 2.5</u>

Create the following LotusScript code for BUTTON5:

```
Sub Click(Source As Button)
        On Error Goto handler
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        Dim session As New lcsession
        Dim src As New lcconnection("db2")
        Dim keys As New lcfieldlist
        Dim fields As New lcfieldlist
        Dim field As lcfield
        Dim empno As lcfield
        Dim lastname As lcfield
        Dim hiredate As lcfield
        session.clearstatus
        src.database="SAMPLE"
        src.userid="Administrator"
        src.password="rac4you"
        src.connect
        src.metadata="EMPLOYEE"
        Set field=keys.append("FIRSTNME",LCTYPE_TEXT)
        field.flags=LCFIELDF_KEY
        field.text="JAMES"
        If (src.select(keys,1,fields)=0) Then
                Messagebox "Selection Error"
                End
        End If
        Set empno=fields.lookup("EMPNO")
        Set lastname=fields.lookup("LASTNAME")
        Set hiredate=fields.lookup("HIREDATE")
        msg1=""
        While (src.fetch(fields)>0)
                msg1=msg1 & "EMPNO= " & empno.text(0) & " LASTNAME= " & lastname.text(0) _
                & " HIREDATE= " & hiredate.text(0) & Chr(10)
        Wend
        Messagebox msg1
        End
handler:
        If (session.status <> LCSUCCESS) Then
                status=session.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
```

```
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.4** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don’t need to save the opened document.
- ✓ Push onto BUTTON5.

Example 2.6

This example produces the same result as **EXAMPLE 2.4** following the same procedure but instead to use FIELDNAMES property of LC_Connection class, it makes use of CATALOG method of LC_Connection class.
Take care to the following bold code:

```
Set catalog=fields.getfield(1)
        While (src.fetch(fields)>0)
                totcatalog=totcatalog  & catalog.text(0) & ","
```

In **set catalog=fields.getfield(1)** code line, it's mandatory to put number 1. **See GetField method of LC_fieldlist class in Domino Release 5. Domino Enterprise Integration Guide book.**

In **totcatalog=totcatalog & catalog.text(0) & ","** code line, you get the names of fields contained in LC_Fieldlist.

In order to achieve this objective do the following step:

Step A - 2.6

Create the following LotusScript code for BUTTON7:

```
Sub Click(Source As Button)
        On Error Goto handler
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        Dim session As New lcsession
        Dim src As New lcconnection("db2")
        Dim keys As New lcfieldlist
        Dim fields As New lcfieldlist
        Dim fields1 As New lcfieldlist
        Dim field As lcfield
        Dim catalog As lcfield
        Dim totcatalog As String
        totcatalog=""
        session.clearstatus
        src.database="SAMPLE"
        src.userid="Administrator"
        src.password="rac4you"
        src.connect
        src.metadata="EMPLOYEE"
        Set field=keys.append("FIRSTNME",LCTYPE_TEXT)
        field.flags=LCFIELDF_KEY
        field.text="JAMES"
        If (src.catalog(LCOBJECT_FIELD,fields)=0) Then
                Messagebox "Error in Catalog"
                End
        End If
        Set catalog=fields.getfield(1)
        While (src.fetch(fields)>0)
```

```
                totcatalog=totcatalog  & catalog.text(0) & ","
        Wend
        If ((Instr(1,totcatalog,"EMPNO",0) <> 0) And (Instr(1,totcatalog,"LASTNAME",0) <> 0) And _
        (Instr(1,totcatalog,"HIREDATE",0) <> 0))Then
                src.fieldnames="EMPNO" & ",LASTNAME,HIREDATE"
        Else
                Messagebox "No ones from the following fields EMPNO, LASTNAME, HIREDATE exist in catalog"
                End
        End If
        If (src.select(keys,1,fields1)=0) Then
                Messagebox "Error in SELECT"
                End
        End If
        msg1=""
        While (src.fetch(fields1)>0)
                msg1=msg1 & "EMPNO= " & fields1.EMPNO(0) & " LASTNAME= " & fields1.LASTNAME(0) _
                & " HIREDATE= " & fields1.HIREDATE(0) & Chr(10)
        Wend
        Messagebox msg1
        End
handler:
        If (session.status <> LCSUCCESS) Then
                status=session.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.6** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON7.

Example 2.7

This example produces the same result as **EXAMPLE 2.4** following the same procedure but instead to use FIELDNAMES property of LC_Connection class, it makes use of MAP method of LC_Connection class.
In order to achieve this objective do the following step:

Step A - 2.7

Create the following LotusScript code for BUTTON20:

```
Sub Click(Source As Button)
        On Error Goto handler
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        Dim session As New lcsession
        Dim src As New lcconnection("db2")
        Dim keys As New lcfieldlist
        Dim fields As New lcfieldlist
        Dim dfield As New lcfieldlist
        Dim field As lcfield
        Dim empno As lcfield
        Dim lastname As lcfield
        Dim hiredate As lcfield
        session.clearstatus
        src.database="SAMPLE"
        src.userid="Administrator"
        src.password="rac4you"
        src.connect
        src.metadata="EMPLOYEE"
        Set field=keys.append("FIRSTNME",LCTYPE_TEXT)
        field.flags=LCFIELDF_KEY
        field.text="JAMES"
        If (src.select(keys,1,fields)=0) Then
                Messagebox "Error in Selection"
                End
        End If
        Call dfield.map(fields,"EMPNO,LASTNAME,HIREDATE")
        Set empno=dfield.getfield(1)
        Set lastname=dfield.getfield(2)
        Set hiredate=dfield.getfield(3)
        src.mapbyname=True
        msg1=""
        While (src.fetch(dfield)>0)
                msg1=msg1 & "EMPNO= " & empno.text(0) & " LASTNAME= " & lastname.text(0) _
                & " HIREDATE= " & hiredate.text(0) & Chr(10)
        Wend
        Messagebox msg1
        End
handler:
        If (session.status <> LCSUCCESS) Then
                status=session.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
```

```
                    & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
          Else
                    Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
          End If
          End
End Sub
```

In order to run **Example 2.7** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON20.

Example 2.8

This example updates a row in EMPLOYEE table for an EMPNO value. It works with the document created in Example 2.1.
Take care to the following bold code:

```
REM The value of a column in DB2 can be set by defining the name of DB/2 column as a
REM PROPERTY of FIELDLIST as is written in the first below line: FldLst.EMPNO="000210"
REM FldLst.EMPNO="000210"
```

In order to achieve the objective of this example, do the following step:

Step A - 2.8

Create the following LotusScript code for BUTTON3:

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim workspace As New notesuiworkspace
        Dim uidoc As notesuidocument
        Dim srccon As New LCConnection("db2")
        Dim FldLst As New LCFieldList(1,LCFIELDF_TRUNC_DATA+LCFIELDF_TRUNC_PREC)
        Dim FirstNmeFld  As New LCField(LCTYPE_TEXT,1)
        Dim EmpNoKeyField As New LCField(LCTYPE_TEXT,1)
        Dim EmpNo As String
        Dim count As Long
        Set uidoc=workspace.currentdocument
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        srcCon.MetaData="EMPLOYEE"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        EmpNo=uidoc.fieldgettext("EmpNoAlias")
        Set EmpNoKeyField=FldLst.append("EMPNO",LCTYPE_TEXT)
        EmpNoKeyField.value=EmpNo

REM The value of a column in DB2 can be set by defining the name of DB/2 column as a
REM PROPERTY of FIELDLIST as is written in the first below line: FldLst.EMPNO="000210"
   REM FldLst.EMPNO="000210"

        Set FirstNmeFld=FldLst.append("FIRSTNME",LCTYPE_TEXT)
        FirstNmeFld.text=uidoc.fieldgettext("FirstNme")
        EmpNoKeyField.Flags=EmpNoKeyField.Flags Or LCFIELDF_KEY
        count=srcCon.update(FldLst,1,1)
        Messagebox  "There are " & count & " record(s) updated"
        Messagebox "Finish Update"
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
```

```
          Dim msgcode As Long
          Dim status As Long
          If (session.status <> LCSUCCESS) Then
                    status=session.getstatus(errortext,msgcode,msg)
                    Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                    & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
          Else
                    Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
          End If
          End
End Sub
```

In order to run **Example 2.8** do the following steps:

- ✓ Open the document, created during Example 2.1, in edit mode.
- ✓ For the name(LASTNAME) displayed in the EmpNo field of document, which has the serial number defined in the field EmpNoAlias of document, we change FIRSTNAME displayed in the field FirstNme of document.
- ✓ Push onto the button BUTTON3.

Example 2.9

This example creates a new table named EUROPE in SAMPLE database. The table will be empty, having the following structure:
CITY, text, 10 chars in size.
COUNTRY, text, 10 chars in size.

Following examples will show how to populate, update and delete records in this table. In order to achieve the objective of this example, do the following step:

<u>Step A - 2.9</u>

Create the following LotusScript code for BUTTON8.

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim srccon As New LCConnection("db2")
        Dim FldLst As New LCFieldList(1,LCFIELDF_TRUNC_DATA+LCFIELDF_TRUNC_PREC)
        Dim fld As lcfield
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        srcCon.MetaData="EUROPE"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        Set fld=FldLst.append("CITY",LCTYPE_TEXT)
        Call fld.setformatstream(0,10,LCSTREAMFMT_NATIVE)
        Set fld=FldLst.append("COUNTRY",LCTYPE_TEXT)
        Call fld.setformatstream(0,10,LCSTREAMFMT_NATIVE)
        Call srcCon.create(LCOBJECT_METADATA,FldLst)
        Messagebox "Finish Creating TABLE"
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
                status=LC_S.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.9** do the following steps:

✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
✓ Push onto BUTTON8.

Example 2.10

This example adds rows into the table created during the **EXAMPLE 2.9**, populating the field CITY with PARIS, and COUNTRY with FRANCE. In the present example, the texts **PARIS** and **FRANCE** are hard coded, but you can build a construction, that asks you to type a specific CITY and COUNTRY respectively. As you can see, many opportunities exist for additional examples here.
Take care to the following remarked code in Step A - 2.10:

REM Instead of the command EmpNoKeyField.value="PARIS" you can use(only for text or binary fields)
REM the following below lines, remarked. Attention !!! You must write all 3 below lines remarked"
REM Dim msgs As New lcstream
REM msgs.text="PARIS"
REM Call EmpNoKeyField.setstream(1,msgs)

In order to achieve the objective of this example, do the following step:

Step A - 2.10

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim uidoc As notesuidocument
        Dim srccon As New LCConnection("db2")
        Dim FldLst As New LCFieldList(1,LCFIELDF_TRUNC_DATA+LCFIELDF_TRUNC_PREC)
        Dim FirstNmeFld  As New LCField(LCTYPE_TEXT,1)
        Dim EmpNoKeyField As New LCField(LCTYPE_TEXT,1)
        Dim count As Long
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        srcCon.MetaData="EUROPE"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        Set EmpNoKeyField=FldLst.append("CITY",LCTYPE_TEXT)
        EmpNoKeyField.value="PARIS"

REM Instead of the command EmpNoKeyField.value="PARIS" you can use(only for text or binary fields)
REM the following below lines, remarked. Attention !!! You must write all 3 below lines remarked"
REM Dim msgs As New lcstream
REM msgs.text="PARIS"
REM Call EmpNoKeyField.setstream(1,msgs)

        Set FirstNmeFld=FldLst.append("COUNTRY",LCTYPE_TEXT)
        FirstNmeFld.text="FRANCE"
        EmpNoKeyField.Flags=EmpNoKeyField.Flags Or LCFIELDF_KEY
        count=srcCon.insert(FldLst,1,1)
        Messagebox  "There are " & count & " record(s) inserted"
        Messagebox "Finish Insertion"
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
```

```
          Dim msgcode As Long
          Dim status As Long
          If (LC_S.status <> LCSUCCESS) Then
                    status=LC_S.getstatus(errortext,msgcode,msg)
                    Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                    & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
          Else
                    Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
          End If
          End
End Sub
```

In order to run **Example 2.10** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON10

Example 2.11

This example deletes all rows into the table, created during the **EXAMPLE 2.9**, for which the column COUNTRY is **FRANCE.** In the present example, the text **FRANCE** is hard coded, but you can build a construction, that asks you to type a specific COUNTRY. As you can see, many opportunities exist for additional examples here.

In order to achieve the objective of this example, do the following step:

<u>Step A - 2.11</u>

Create the following LotusScript code for BUTTON11:

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim srccon As New LCConnection("db2")
        Dim FldLst As New LCFieldList(1,LCFIELDF_TRUNC_DATA+LCFIELDF_TRUNC_PREC)
        Dim tara As lcfield
        Dim count As Long
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        srcCon.MetaData="EUROPE"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        srcCon.mapbyname=True
        Set tara=FldLst.append("COUNTRY",LCTYPE_TEXT)
        tara.value="FRANCE"
        tara.Flags=tara.Flags Or LCFIELDF_KEY
        count=srcCon.remove(FldLst,1,1)
        Messagebox  "There are " & count & " record(s) Deleted"
        Messagebox "Finish Delete Records"
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
                status=LC_S.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.11** do the following steps:

✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.

- ✓ Push onto BUTTON11.

Example 2.12

This example removes, using the method DROP of LC_Connection class, the table created during the **EXAMPLE 2.9**.
In order to achieve this objective do the following step:

Step A - 2.12

Create the following LotusScript code for BUTTON9:

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim srccon As New LCConnection("db2")
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        srcCon.MetaData="EUROPE"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        Call srcCon.drop(LCOBJECT_METADATA)
        Messagebox "Finish Delete TABLE"
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
                status=LC_S.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.12** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON9.

Example 2.13

This example retrieves a copy of the current value for a connection property. Actually it shows the values behind Property Token from **Appendix B** of **Domino Release 5. Domino Enterprise Integration Guide** book.
In order to achieve this objective do the following step:

Step A - 2.13

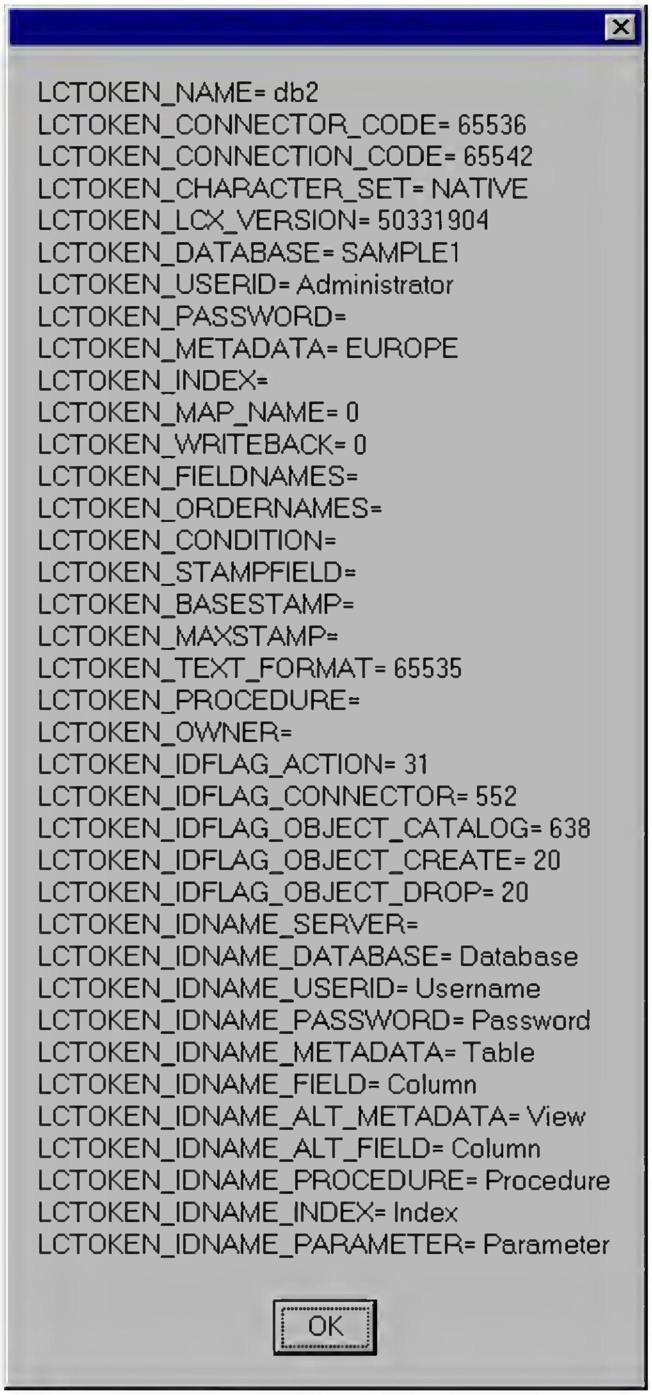
Create the following LotusScript code for BUTTON9:

```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim srccon As New LCConnection("db2")
        Dim FldLst As New LCFieldList(1,LCFIELDF_TRUNC_DATA+LCFIELDF_TRUNC_PREC)
        Dim nume, connector_code, connection_code, character_set, lcx_version  As lcfield
        Dim database, userid, password, metadata, index, map_name, writeback, fieldnames, ordernames As lcfield
        Dim condition, stampfield, basestamp, maxstamp, text_format, procedure, owner As lcfield
        Dim idflag_action, idflag_connector, idflag_object_catalog, idflag_object_create, idflag_object_drop As lcfield
        Dim idname_server, idname_database, idname_userid, idname_password, idname_metadata, idname_field As lcfield
        Dim idname_alt_metadata, idname_alt_field, idname_procedure, idname_index, idname_parameter As lcfield
        Dim count As Long
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        srcCon.MetaData="EUROPE"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        Set nume=srcCon.getproperty(LCTOKEN_NAME)
        Set connector_code=srcCon.getproperty(LCTOKEN_CONNECTOR_CODE)
        Set connection_code=srcCon.getproperty(LCTOKEN_CONNECTION_CODE)
        Set character_text=srcCon.getproperty(LCTOKEN_CHARACTER_SET)
        Set lcx_version=srcCon.getproperty(LCTOKEN_LCX_VERSION)
        Set database=srcCon.getproperty(LCTOKEN_DATABASE)
        Set userid=srcCon.getproperty(LCTOKEN_USERID)
        Set password=srcCon.getproperty(LCTOKEN_PASSWORD)
        Set metadata=srcCon.getproperty(LCTOKEN_METADATA)
        Set index=srcCon.getproperty(LCTOKEN_INDEX)
        Set map_name=srcCon.getproperty(LCTOKEN_MAP_NAME)
        Set writeback=srcCon.getproperty(LCTOKEN_WRITEBACK)
        Set fieldnames=srcCon.getproperty(LCTOKEN_FIELDNAMES)
        Set ordernames=srcCon.getproperty(LCTOKEN_ORDERNAMES)
        Set condition=srcCon.getproperty(LCTOKEN_CONDITION)
        Set stampfield=srcCon.getproperty(LCTOKEN_STAMPFIELD)
        Set basestamp=srcCon.getproperty(LCTOKEN_BASESTAMP)
        Set maxstamp=srcCon.getproperty(LCTOKEN_MAXSTAMP)
        Set text_format=srcCon.getproperty(LCTOKEN_TEXT_FORMAT)
        Set procedure=srcCon.getproperty(LCTOKEN_PROCEDURE)
        Set owner=srcCon.getproperty(LCTOKEN_OWNER)
        Set idflag_action=srcCon.getproperty(LCTOKEN_IDFLAG_ACTION)
        Set idflag_connector=srcCon.getproperty(LCTOKEN_IDFLAG_CONNECTOR)
        Set idflag_object_catalog=srcCon.getproperty(LCTOKEN_IDFLAG_OBJECT_CATALOG)
        Set idflag_object_create=srcCon.getproperty(LCTOKEN_IDFLAG_OBJECT_CREATE)
        Set idflag_object_drop=srcCon.getproperty(LCTOKEN_IDFLAG_OBJECT_DROP)
        Set idname_server=srcCon.getproperty(LCTOKEN_IDNAME_SERVER)
        Set idname_database=srcCon.getproperty(LCTOKEN_IDNAME_DATABASE)
        Set idname_userid=srcCon.getproperty(LCTOKEN_IDNAME_USERID)
        Set idname_password=srcCon.getproperty(LCTOKEN_IDNAME_PASSWORD)
        Set idname_metadata=srcCon.getproperty(LCTOKEN_IDNAME_METADATA)
        Set idname_field=srcCon.getproperty(LCTOKEN_IDNAME_FIELD)
        Set idname_alt_metadata=srcCon.getproperty(LCTOKEN_IDNAME_ALT_METADATA)
        Set idname_alt_field=srcCon.getproperty(LCTOKEN_IDNAME_ALT_FIELD)
        Set idname_procedure=srcCon.getproperty(LCTOKEN_IDNAME_PROCEDURE)
        Set idname_index=srcCon.getproperty(LCTOKEN_IDNAME_INDEX)
        Set idname_parameter=srcCon.getproperty(LCTOKEN_IDNAME_PARAMETER)
        Messagebox "LCTOKEN_NAME= " & nume.text(0) & Chr(10) _
        & "LCTOKEN_CONNECTOR_CODE= " & connector_code.text(0) & Chr(10) _
        & "LCTOKEN_CONNECTION_CODE= " &  connection_code.text(0) & Chr(10) _
        & "LCTOKEN_CHARACTER_SET= " &  character_text.text(0) & Chr(10) _
        & "LCTOKEN_LCX_VERSION= " &  lcx_version.text(0) & Chr(10) _
        & "LCTOKEN_DATABASE= " & database.text(0) & Chr(10) _
        & "LCTOKEN_USERID= "  & userid.text(0) & Chr(10) _
        & "LCTOKEN_PASSWORD= " & password.text(0) & Chr(10) _
        & "LCTOKEN_METADATA= " & metadata.text(0) & Chr(10) _
        & "LCTOKEN_INDEX= " & index.text(0) & Chr(10) _
        & "LCTOKEN_MAP_NAME= " & map_name.text(0) & Chr(10) _
        & "LCTOKEN_WRITEBACK= " & writeback.text(0) & Chr(10) _
        & "LCTOKEN_FIELDNAMES= " & fieldnames.text(0) & Chr(10) _
        & "LCTOKEN_ORDERNAMES= " & ordernames.text(0) & Chr(10) _
        & "LCTOKEN_CONDITION= " & condition.text(0) & Chr(10) _
        & "LCTOKEN_STAMPFIELD= " & stampfield.text(0) & Chr(10) _
        & "LCTOKEN_BASESTAMP= " & basestamp.text(0) & Chr(10) _
        & "LCTOKEN_MAXSTAMP= " & maxstamp.text(0) & Chr(10) _
        & "LCTOKEN_TEXT_FORMAT= " & text_format.text(0) & Chr(10) _
        & "LCTOKEN_PROCEDURE= " & procedure.text(0) & Chr(10) _
        & "LCTOKEN_OWNER= " & owner.text(0) & Chr(10) _
        & "LCTOKEN_IDFLAG_ACTION= " & idflag_action.text(0) & Chr(10) _
        & "LCTOKEN_IDFLAG_CONNECTOR= " & idflag_connector.text(0) & Chr(10) _
        & "LCTOKEN_IDFLAG_OBJECT_CATALOG= " & idflag_object_catalog.text(0) & Chr(10) _
        & "LCTOKEN_IDFLAG_OBJECT_CREATE= " & idflag_object_create.text(0) & Chr(10) _
        & "LCTOKEN_IDFLAG_OBJECT_DROP= " & idflag_object_drop.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_SERVER= " & idname_server.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_DATABASE= " & idname_database.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_USERID= " & idname_userid.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_PASSWORD= " & idname_password.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_METADATA= " &  idname_metadata.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_FIELD= " & idname_field.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_ALT_METADATA= " & idname_alt_metadata.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_ALT_FIELD= " &  idname_alt_field.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_PROCEDURE= " &  idname_procedure.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_INDEX= " &  idname_index.text(0) & Chr(10) _
        & "LCTOKEN_IDNAME_PARAMETER= " &  idname_parameter.text(0)
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
                status=LC_S.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.13** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON12

The result is as follows:



```
LCTOKEN_NAME= db2
LCTOKEN_CONNECTOR_CODE= 65536
LCTOKEN_CONNECTION_CODE= 65542
LCTOKEN_CHARACTER_SET= NATIVE
LCTOKEN_LCX_VERSION= 50331904
LCTOKEN_DATABASE= SAMPLE1
LCTOKEN_USERID= Administrator
LCTOKEN_PASSWORD=
LCTOKEN_METADATA= EUROPE
LCTOKEN_INDEX=
LCTOKEN_MAP_NAME= 0
LCTOKEN_WRITEBACK= 0
LCTOKEN_FIELDNAMES=
LCTOKEN_ORDERNAMES=
LCTOKEN_CONDITION=
LCTOKEN_STAMPFIELD=
LCTOKEN_BASESTAMP=
LCTOKEN_MAXSTAMP=
LCTOKEN_TEXT_FORMAT= 65535
LCTOKEN_PROCEDURE=
LCTOKEN_OWNER=
LCTOKEN_IDFLAG_ACTION= 31
LCTOKEN_IDFLAG_CONNECTOR= 552
LCTOKEN_IDFLAG_OBJECT_CATALOG= 638
LCTOKEN_IDFLAG_OBJECT_CREATE= 20
LCTOKEN_IDFLAG_OBJECT_DROP= 20
LCTOKEN_IDNAME_SERVER=
LCTOKEN_IDNAME_DATABASE= Database
LCTOKEN_IDNAME_USERID= Username
LCTOKEN_IDNAME_PASSWORD= Password
LCTOKEN_IDNAME_METADATA= Table
LCTOKEN_IDNAME_FIELD= Column
LCTOKEN_IDNAME_ALT_METADATA= View
LCTOKEN_IDNAME_ALT_FIELD= Column
LCTOKEN_IDNAME_PROCEDURE= Procedure
LCTOKEN_IDNAME_INDEX= Index
LCTOKEN_IDNAME_PARAMETER= Parameter
```

Example 2.14

This example retrieves all properties supported by a connector. Actually it shows the values behind Property Token from **Appendix B** and **Appendix C** of **Domino Release 5. Domino Enterprise Integration Guide** book.
In order to achieve this objective do the following step:

Step A - 2.14

Create the following LotusScript code for BUTTON13:

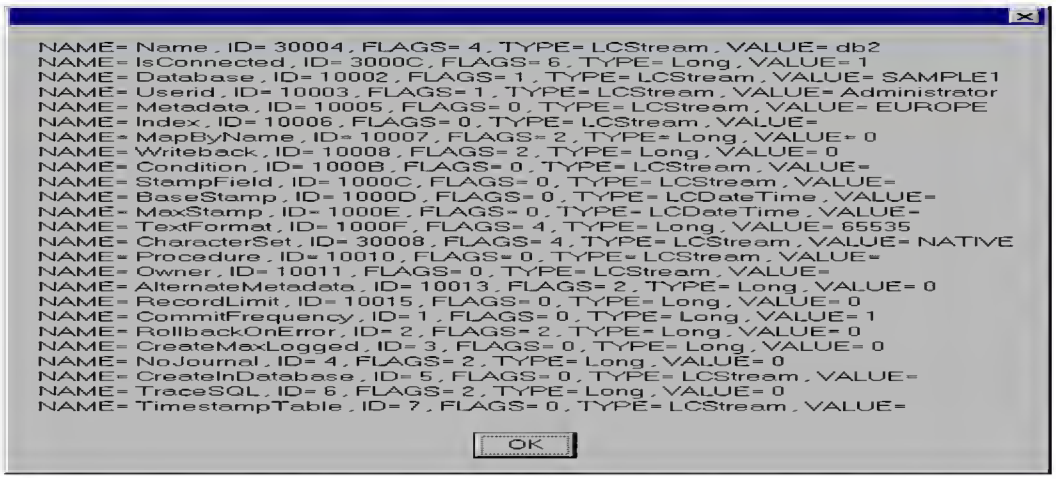
```
Sub Click(Source As Button)
	Dim LC_S As New LCSession
	Dim SrcCon As New LCConnection("db2")
	Dim confld As lcfield
	Dim propname As String
	Dim propdate As lcdatetime
	Dim propnumeric As lcnumeric
	Dim propstrm As lcstream
	Dim propcurr As lccurrency
	Dim propfloat As Double
	Dim propint As Long
	Dim propbool As Variant
	Dim tokenid As Long
	Dim proptype As Long
	Dim propflags As Long
	On Error Goto ErrorHandler
	SrcCon.Userid="Administrator"
	SrcCon.Password="rac4you"
	SrcCon.Database="SAMPLE"
	srcCon.MetaData="EUROPE"
	srcCon.Disconnect
	LC_S.ClearStatus
	srcCon.Connect
	Call SrcCon.listproperty(LCLIST_FIRST, tokenid, proptype, propflags, propname)
	msg1=""
	Do
		Set confld=SrcCon.getproperty(tokenid)
		Select Case proptype
		Case LCTYPE_DATETIME:
			Set propdate=SrcCon.getpropertydatetime(tokenid)
			msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "LCDateTime" & " , VALUE= " & propdate.text & Chr(10)
		Case LCTYPE_NUMERIC:
			Set propnumeric=SrcCon.getpropertynumeric(tokenid)
			msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "LCNumeric" & " , VALUE= " & propnumeric.text & Chr(10)
		Case LCTYPE_TEXT:
			Set propstrm=SrcCon.getpropertystream(tokenid,LCSTREAMFMT_NATIVE)
			msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "LCStream" & " , VALUE= " & propstrm.text & Chr(10)
		Case LCTYPE_CURRENCY:
			Set propcurr=SrcCon.getpropertycurrency(tokenid)
			msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "LCCurrency" & " , VALUE= " & propcurr.text & Chr(10)
		Case LCTYPE_FLOAT:
			propfloat=SrcCon.getpropertyfloat(tokenid)
```

```
                msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "Double" & " , VALUE= " & Cstr(propfloat) & Chr(10)
            Case LCTYPE_INT:
                If (propflags And LCPROPERTY_BOOLEAN) Then
                    propbool=SrcCon.getpropertyboolean(tokenid,False)
                    msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "Boolean" & " , VALUE= " & Cstr(propbool) & Chr(10)
                Else
                    propint=SrcCon.getpropertyint(tokenid)
                    msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " &
Hex(propflags) & " , TYPE= " & "Long" & " , VALUE= " & Cstr(propint)     & Chr(10)
                End If
            End Select
        Loop While SrcCon.listproperty(LCLIST_NEXT, tokenid, proptype, propflags, propname)
        Messagebox msg1
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
            status=LC_S.getstatus(errortext,msgcode,msg)
            Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
            & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
            Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.14** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON13

The result is as follows:



```
NAME= Name , ID= 30004 , FLAGS= 4 , TYPE= LCStream , VALUE= db2
NAME= IsConnected , ID= 3000C , FLAGS= 6 , TYPE= Long , VALUE= 1
NAME= Database , ID= 10002 , FLAGS= 1 , TYPE= LCStream , VALUE= SAMPLE1
NAME= Userid , ID= 10003 , FLAGS= 1 , TYPE= LCStream , VALUE= Administrator
NAME= Metadata , ID= 10005 , FLAGS= 0 , TYPE= LCStream , VALUE= EUROPE
NAME= Index , ID= 10006 , FLAGS= 0 , TYPE= LCStream , VALUE=
NAME= MapByName , ID= 10007 , FLAGS= 2 , TYPE= Long , VALUE= 0
NAME= Writeback , ID= 10008 , FLAGS= 2 , TYPE= Long , VALUE= 0
NAME= Condition , ID= 1000B , FLAGS= 0 , TYPE= LCStream , VALUE=
NAME= StampField , ID= 1000C , FLAGS= 0 , TYPE= LCStream , VALUE=
NAME= BaseStamp , ID= 1000D , FLAGS= 0 , TYPE= LCDateTime , VALUE=
NAME= MaxStamp , ID= 1000E , FLAGS= 0 , TYPE= LCDateTime , VALUE=
NAME= TextFormat , ID= 1000F , FLAGS= 4 , TYPE= Long , VALUE= 65535
NAME= CharacterSet , ID= 30008 , FLAGS= 4 , TYPE= LCStream , VALUE= NATIVE
NAME= Procedure , ID= 10010 , FLAGS= 0 , TYPE= LCStream , VALUE=
NAME= Owner , ID= 10011 , FLAGS= 0 , TYPE= LCStream , VALUE=
NAME= AlternateMetadata , ID= 10013 , FLAGS= 2 , TYPE= Long , VALUE= 0
NAME= RecordLimit , ID= 10015 , FLAGS= 0 , TYPE= Long , VALUE= 0
NAME= CommitFrequency , ID= 1 , FLAGS= 0 , TYPE= Long , VALUE= 1
NAME= RollbackOnError , ID= 2 , FLAGS= 2 , TYPE= Long , VALUE= 0
NAME= CreateMaxLogged , ID= 3 , FLAGS= 0 , TYPE= Long , VALUE= 0
NAME= NoJournal , ID= 4 , FLAGS= 2 , TYPE= Long , VALUE= 0
NAME= CreateInDatabase , ID= 5 , FLAGS= 0 , TYPE= LCStream , VALUE=
NAME= TraceSQL , ID= 6 , FLAGS= 2 , TYPE= Long , VALUE= 0
NAME= TimestampTable , ID= 7 , FLAGS= 0 , TYPE= LCStream , VALUE=
```

Example 2.15

This example produces the same result as **Example 2.14** but brings -up more details about all properties supported by a connector.
In order to achieve this objective do the following step:

Step A - 2.15

Create the following LotusScript code for BUTTON14:

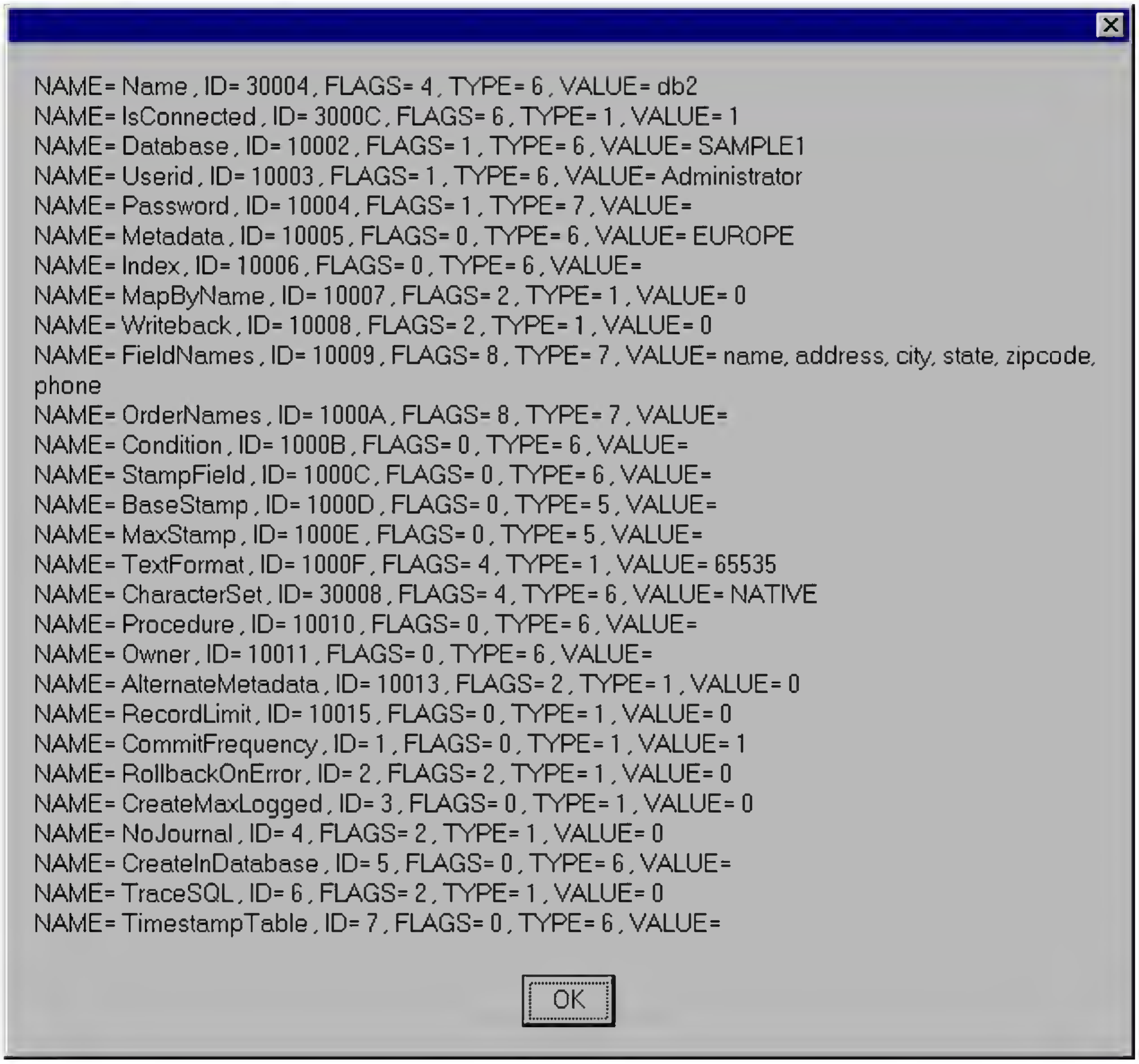
```
Sub Click(Source As Button)
        Dim LC_S As New LCSession
        Dim SrcCon As New LCConnection("db2")
        Dim confld As lcfield
        Dim propname As String
        Dim tokenid As Long
        Dim proptype As Long
        Dim propflags As Long
        On Error Goto ErrorHandler
        SrcCon.Userid="Administrator"
        SrcCon.Password="rac4you"
        SrcCon.Database="SAMPLE"
        SrcCon.MetaData="EUROPE"
        SrcCon.fieldnames="name, address, city, state, zipcode, phone"
        srcCon.Disconnect
        LC_S.ClearStatus
        srcCon.Connect
        Call SrcCon.listproperty(LCLIST_FIRST, tokenid, proptype, propflags, propname)
        msg1=""
        Do
                Set confld=SrcCon.getproperty(tokenid)
                msg1=msg1 & "NAME= " & propname & " , ID= " & Hex(tokenid) & " , FLAGS= " & Hex(propflags) & " ,
TYPE= " & proptype & " , VALUE= " & confld.text(0) & Chr(10)
        Loop While SrcCon.listproperty(LCLIST_NEXT, tokenid, proptype, propflags, propname)
        Messagebox msg1
        srcCon.disconnect
        End
ErrorHandler:
        Messagebox "Attention ! You are in Error"
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim status As Long
        If (LC_S.status <> LCSUCCESS) Then
                status=LC_S.getstatus(errortext,msgcode,msg)
                Messagebox "Internal Error Text= " & errortext & Chr(10) & "Internal Error Code= " & status & Chr(10) _
                & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
        Else
                Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
        End If
        End
End Sub
```

In order to run **Example 2.15** do the following steps:

✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.

✓ Push onto BUTTON14

The result is as follows:



```
NAME= Name , ID= 30004 , FLAGS= 4 , TYPE= 6 , VALUE= db2
NAME= IsConnected , ID= 3000C , FLAGS= 6 , TYPE= 1 , VALUE= 1
NAME= Database , ID= 10002 , FLAGS= 1 , TYPE= 6 , VALUE= SAMPLE1
NAME= Userid , ID= 10003 , FLAGS= 1 , TYPE= 6 , VALUE= Administrator
NAME= Password , ID= 10004 , FLAGS= 1 , TYPE= 7 , VALUE=
NAME= Metadata , ID= 10005 , FLAGS= 0 , TYPE= 6 , VALUE= EUROPE
NAME= Index , ID= 10006 , FLAGS= 0 , TYPE= 6 , VALUE=
NAME= MapByName , ID= 10007 , FLAGS= 2 , TYPE= 1 , VALUE= 0
NAME= Writeback , ID= 10008 , FLAGS= 2 , TYPE= 1 , VALUE= 0
NAME= FieldNames , ID= 10009 , FLAGS= 8 , TYPE= 7 , VALUE= name, address, city, state, zipcode,
phone
NAME= OrderNames , ID= 1000A , FLAGS= 8 , TYPE= 7 , VALUE=
NAME= Condition , ID= 1000B , FLAGS= 0 , TYPE= 6 , VALUE=
NAME= StampField , ID= 1000C , FLAGS= 0 , TYPE= 6 , VALUE=
NAME= BaseStamp , ID= 1000D , FLAGS= 0 , TYPE= 5 , VALUE=
NAME= MaxStamp , ID= 1000E , FLAGS= 0 , TYPE= 5 , VALUE=
NAME= TextFormat , ID= 1000F , FLAGS= 4 , TYPE= 1 , VALUE= 65535
NAME= CharacterSet , ID= 30008 , FLAGS= 4 , TYPE= 6 , VALUE= NATIVE
NAME= Procedure , ID= 10010 , FLAGS= 0 , TYPE= 6 , VALUE=
NAME= Owner , ID= 10011 , FLAGS= 0 , TYPE= 6 , VALUE=
NAME= AlternateMetadata , ID= 10013 , FLAGS= 2 , TYPE= 1 , VALUE= 0
NAME= RecordLimit , ID= 10015 , FLAGS= 0 , TYPE= 1 , VALUE= 0
NAME= CommitFrequency , ID= 1 , FLAGS= 0 , TYPE= 1 , VALUE= 1
NAME= RollbackOnError , ID= 2 , FLAGS= 2 , TYPE= 1 , VALUE= 0
NAME= CreateMaxLogged , ID= 3 , FLAGS= 0 , TYPE= 1 , VALUE= 0
NAME= NoJournal , ID= 4 , FLAGS= 2 , TYPE= 1 , VALUE= 0
NAME= CreateInDatabase , ID= 5 , FLAGS= 0 , TYPE= 6 , VALUE=
NAME= TraceSQL , ID= 6 , FLAGS= 2 , TYPE= 1 , VALUE= 0
NAME= TimestampTable , ID= 7 , FLAGS= 0 , TYPE= 6 , VALUE=
```

Example 2.16

This example passes through all valid connectors of a Lotus Extension for Lotus Connectors installation. It gives you information from a Lotus Connector about its supported functionality and naming used by the backend systems as well as the sort of Flags supported by LC_Stream class
In order to achieve this objective do the following step:

Step A - 2.16

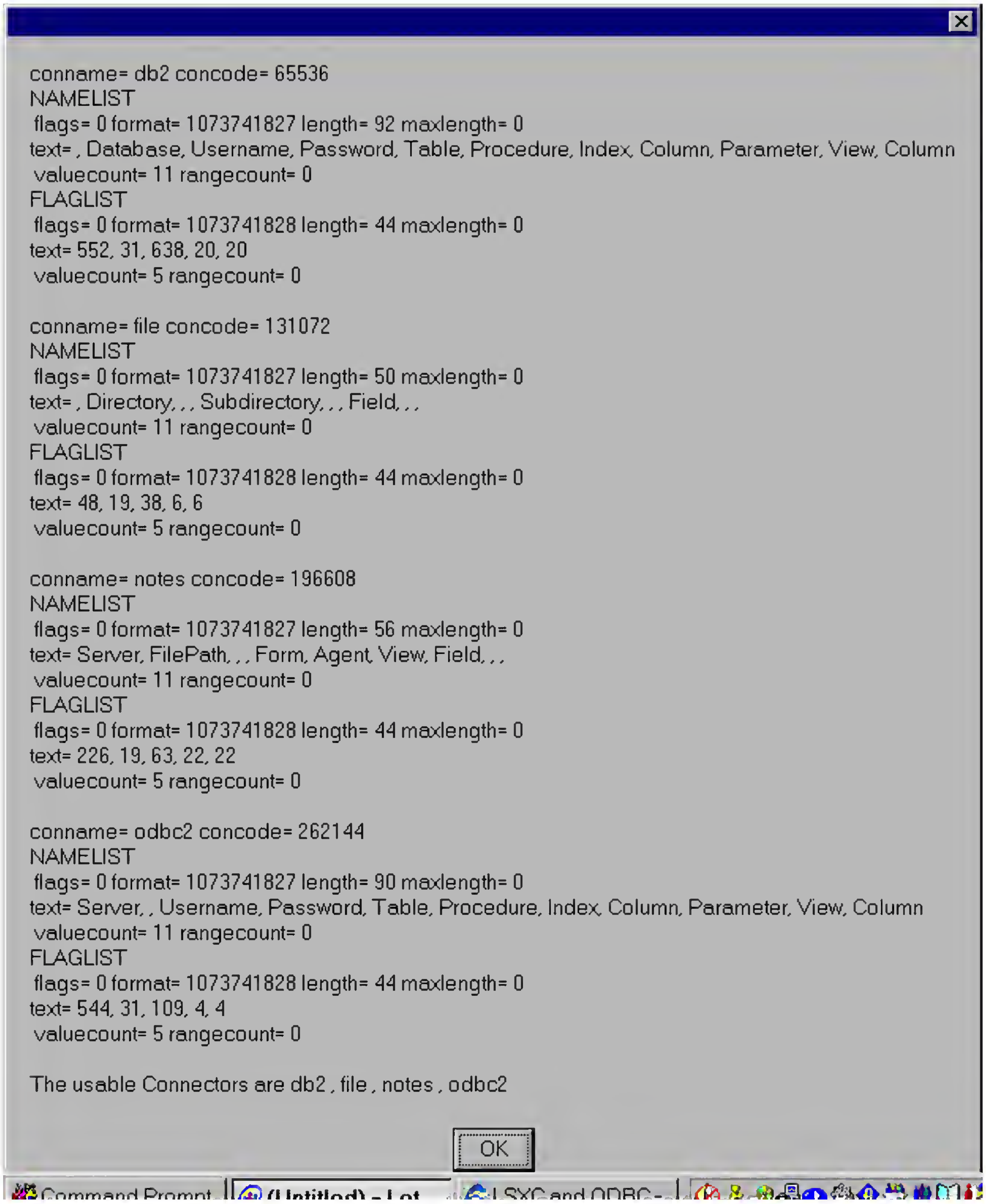
Create the following LotusScript code for BUTTON15:

```
Sub Click(Source As Button)
	Dim session As New lcsession
	Dim conname As String
	Dim concode As Long
	Dim text As String
	Dim flaglist As New lcstream(0,0,LCSTREAMFMT_NUMBER_LIST)
	Dim namelist As  New lcstream(0,0,LCSTREAMFMT_TEXT_LIST)
	Call session.listconnector(LCLIST_FIRST,conname, concode, flaglist, namelist)
	text=conname
	msg1=""
	msg1=msg1 & "conname= " & conname & " concode= " & concode & Chr(10) _
	& "NAMELIST" & Chr(10) _
	& " flags= " & namelist.flags & " format= " & namelist.format _
	& " length= " & namelist.length & " maxlength= " & namelist.maxlength & Chr(10) & "text= " & namelist.text &
Chr(10) _
	& " valuecount= " & namelist.valuecount & " rangecount= " & namelist.rangecount & Chr(10) _
	& "FLAGLIST" & Chr(10) _
	& " flags= " & flaglist.flags & " format= " & flaglist.format _
	& " length= " & flaglist.length & " maxlength= " & flaglist.maxlength & Chr(10) & "text= " & flaglist.text &
Chr(10) _
	& " valuecount= " & flaglist.valuecount & " rangecount= " & flaglist.rangecount & Chr(10) & Chr(10)
	While session.listconnector(LCLIST_NEXT,conname, concode, flaglist, namelist)
		text=text + " , " + conname
		msg1=msg1 & "conname= " & conname & " concode= " & concode & Chr(10) _
		& "NAMELIST" & Chr(10) _
		& " flags= " & namelist.flags & " format= " & namelist.format _
		& " length= " & namelist.length & " maxlength= " & namelist.maxlength & Chr(10) & "text= " &
namelist.text & Chr(10) _
		& " valuecount= " & namelist.valuecount & " rangecount= " & namelist.rangecount & Chr(10) _
		& "FLAGLIST" & Chr(10) _
		& " flags= " & flaglist.flags & " format= " & flaglist.format _
		& " length= " & flaglist.length & " maxlength= " & flaglist.maxlength & Chr(10) & "text= " & flaglist.text
& Chr(10) _
		& " valuecount= " & flaglist.valuecount & " rangecount= " & flaglist.rangecount & Chr(10) & Chr(10)
	Wend
	msg1=msg1 & "The usable Connectors are " & text
	Messagebox msg1
End Sub
```

In order to run **Example 2.16** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON15

The result is as follows:



```
conname= db2 concode= 65536
NAMELIST
 flags= 0 format= 1073741827 length= 92 maxlength= 0
text= , Database, Username, Password, Table, Procedure, Index, Column, Parameter, View, Column
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 552, 31, 638, 20, 20
 valuecount= 5 rangecount= 0

conname= file concode= 131072
NAMELIST
 flags= 0 format= 1073741827 length= 50 maxlength= 0
text= , Directory, , , Subdirectory, , , Field, , ,
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 48, 19, 38, 6, 6
 valuecount= 5 rangecount= 0

conname= notes concode= 196608
NAMELIST
 flags= 0 format= 1073741827 length= 56 maxlength= 0
text= Server, FilePath, , , Form, Agent, View, Field, , ,
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 226, 19, 63, 22, 22
 valuecount= 5 rangecount= 0

conname= odbc2 concode= 262144
NAMELIST
 flags= 0 format= 1073741827 length= 90 maxlength= 0
text= Server, , Username, Password, Table, Procedure, Index, Column, Parameter, View, Column
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 544, 31, 109, 4, 4
 valuecount= 5 rangecount= 0

The usable Connectors are db2 , file , notes , odbc2
```

Example 2.17

This example passes through all valid MetaConnectors of a Lotus Extension for Lotus Connectors installation. It gives you information from a Lotus Connector about its supported functionality and naming used by the backend systems as well as the sort of Flags supported by LC_Stream class
In order to achieve this objective do the following step:

Step A - 2.17

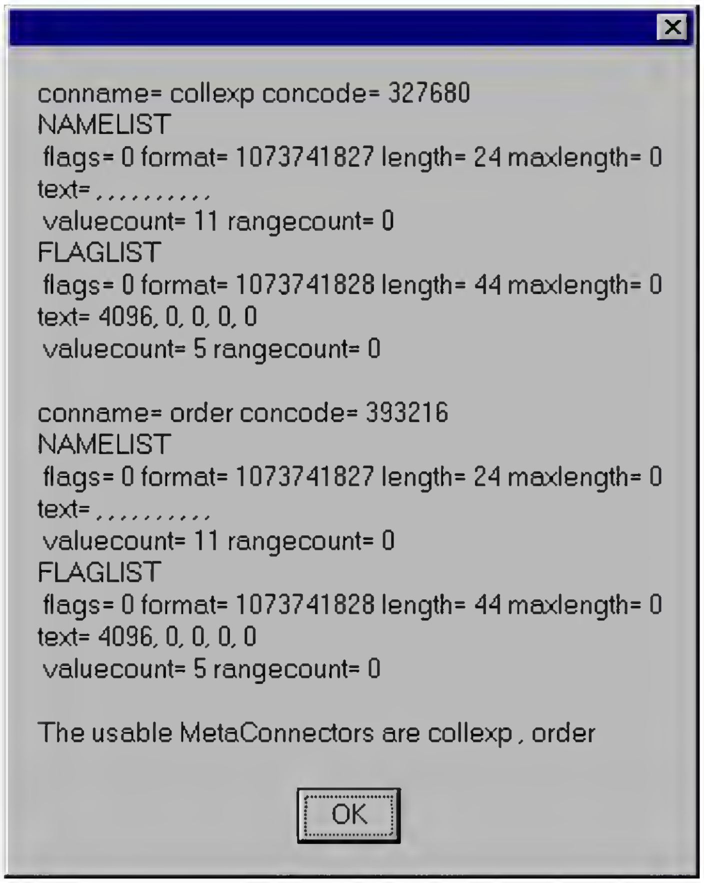
Create the following LotusScript code for BUTTON16:

```
Sub Click(Source As Button)
        Dim session As New lcsession
        Dim conname As String
        Dim concode As Long
        Dim text As String
        Dim flaglist As New lcstream(0,0,LCSTREAMFMT_NUMBER_LIST)
        Dim namelist As  New lcstream(0,0,LCSTREAMFMT_TEXT_LIST)
        Call session.listmetaconnector(LCLIST_FIRST,conname,concode, flaglist, namelist)
        text=conname
        msg1=""
        msg1=msg1 & "conname= " & conname & " concode= " & concode & Chr(10) _
        & "NAMELIST" & Chr(10) _
        & " flags= " & namelist.flags & " format= " & namelist.format _
        & " length= " & namelist.length & " maxlength= " & namelist.maxlength & Chr(10) & "text= " & namelist.text &
Chr(10) _
        & " valuecount= " & namelist.valuecount & " rangecount= " & namelist.rangecount & Chr(10) _
        & "FLAGLIST" & Chr(10) _
        & " flags= " & flaglist.flags & " format= " & flaglist.format _
        & " length= " & flaglist.length & " maxlength= " & flaglist.maxlength & Chr(10) & "text= " & flaglist.text & Chr(10) _
        & " valuecount= " & flaglist.valuecount & " rangecount= " & flaglist.rangecount & Chr(10) & Chr(10)
        While session.listmetaconnector(LCLIST_NEXT,conname,concode, flaglist, namelist)
                text=text + " , " + conname
                msg1=msg1 & "conname= " & conname & " concode= " & concode & Chr(10) _
                & "NAMELIST" & Chr(10) _
                & " flags= " & namelist.flags & " format= " & namelist.format _
                & " length= " & namelist.length & " maxlength= " & namelist.maxlength & Chr(10) & "text= " &
namelist.text & Chr(10) _
                & " valuecount= " & namelist.valuecount & " rangecount= " & namelist.rangecount & Chr(10) _
                & "FLAGLIST" & Chr(10) _
                & " flags= " & flaglist.flags & " format= " & flaglist.format _
                & " length= " & flaglist.length & " maxlength= " & flaglist.maxlength & Chr(10) & "text= " & flaglist.text
& Chr(10) _
                & " valuecount= " & flaglist.valuecount & " rangecount= " & flaglist.rangecount & Chr(10) & Chr(10)
        Wend
        msg1=msg1 & "The usable MetaConnectors are " & text
        Messagebox msg1
End Sub
```

In order to run **Example 2.17** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON16

The result is as follows:



```
conname= collexp concode= 327680
NAMELIST
 flags= 0 format= 1073741827 length= 24 maxlength= 0
text= , , , , , , , , , ,
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 4096, 0, 0, 0, 0
 valuecount= 5 rangecount= 0

conname= order concode= 393216
NAMELIST
 flags= 0 format= 1073741827 length= 24 maxlength= 0
text= , , , , , , , , , ,
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 4096, 0, 0, 0, 0
 valuecount= 5 rangecount= 0

The usable MetaConnectors are collexp , order
```

Example 2.18

This example looks up a Connector name, gives all its features as well as the sort of Flags supported by LC_Stream class
In order to achieve this objective do the following step:

Step A - 2.18

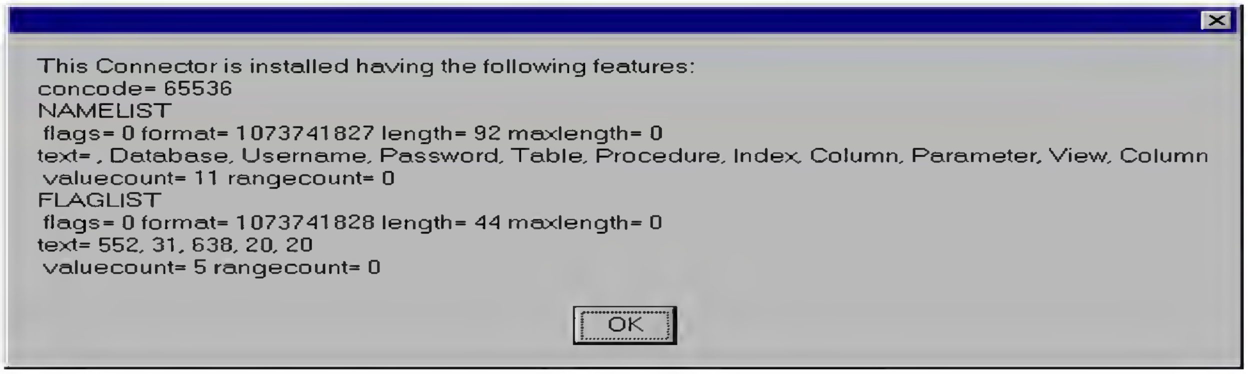
Create the following LotusScript code for BUTTON17:

```
Sub Click(Source As Button)
        Dim session As New lcsession
        Dim concode As Long
        Dim flaglist As New lcstream(0,0,LCSTREAMFMT_NUMBER_LIST)
        Dim namelist As  New lcstream(0,0,LCSTREAMFMT_TEXT_LIST)
        If (session.lookupconnector("db2",concode,flaglist, namelist)) Then
                Messagebox "This Connector is installed having the following features:" & Chr(10) _
                & "concode= " & concode & Chr(10) _
                &"NAMELIST" & Chr(10) _
                & " flags= " & namelist.flags & " format= " & namelist.format _
                & " length= " & namelist.length & " maxlength= " & namelist.maxlength & Chr(10) _
                & "text= " & namelist.text & Chr(10) _
                & " valuecount= " & namelist.valuecount & " rangecount= " & namelist.rangecount & Chr(10) _
                & "FLAGLIST" & Chr(10) _
                & " flags= " & flaglist.flags & " format= " & flaglist.format _
                & " length= " & flaglist.length & " maxlength= " & flaglist.maxlength & Chr(10) _
                & "text= " & flaglist.text & Chr(10) _
                & " valuecount= " & flaglist.valuecount & " rangecount= " & flaglist.rangecount
        Else
                Messagebox "This Connector is not installed"
        End If
End Sub
```

In order to run **Example 2.18** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON17

The result is as follows:



```
This Connector is installed having the following features:
concode= 65536
NAMELIST
 flags= 0 format= 1073741827 length= 92 maxlength= 0
text= , Database, Username, Password, Table, Procedure, Index, Column, Parameter, View, Column
 valuecount= 11 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741828 length= 44 maxlength= 0
text= 552, 31, 638, 20, 20
 valuecount= 5 rangecount= 0
```

Example 2.19

This example looks up a MetaConnector name, gives all its features as well as the sort of Flags supported by LC_Stream class
In order to achieve this objective do the following step:

Step A - 2.18

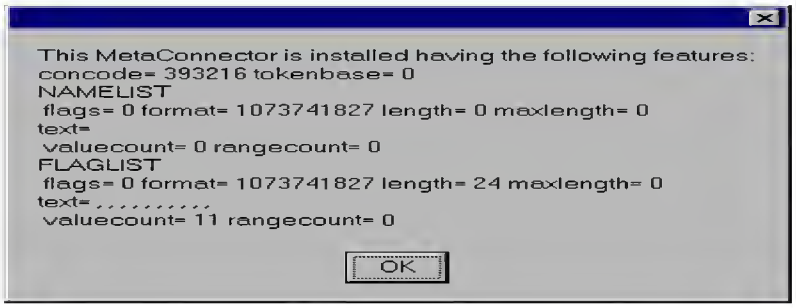
Create the following LotusScript code for BUTTON18:

```
Sub Click(Source As Button)
        Dim session As New lcsession
        Dim concode As Long
        Dim tokenbase As Long
        Dim flaglist As New lcstream(0,0,LCSTREAMFMT_NUMBER_LIST)
        Dim namelist As  New lcstream(0,0,LCSTREAMFMT_TEXT_LIST)
        If (session.lookupmetaconnector("order",concode,tokenbase, flaglist, namelist)) Then
                Messagebox "This MetaConnector is installed having the following features:" & Chr(10) _
                & "concode= " & concode & " tokenbase= " & tokenbase & Chr(10) _
                &"NAMELIST" & Chr(10) _
                & " flags= " & namelist.flags & " format= " & namelist.format _
                & " length= " & namelist.length & " maxlength= " & namelist.maxlength & Chr(10) _
                & "text= " & namelist.text & Chr(10) _
                & " valuecount= " & namelist.valuecount & " rangecount= " & namelist.rangecount & Chr(10) _
                & "FLAGLIST" & Chr(10) _
                & " flags= " & flaglist.flags & " format= " & flaglist.format _
                & " length= " & flaglist.length & " maxlength= " & flaglist.maxlength & Chr(10) _
                & "text= " & flaglist.text & Chr(10) _
                & " valuecount= " & flaglist.valuecount & " rangecount= " & flaglist.rangecount
        Else
                Messagebox "This MetaConnector is not installed"
        End If
End Sub
```

In order to run **Example 2.19** do the following steps:

- ✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.
- ✓ Push onto BUTTON18

The result is as follows:



```
This MetaConnector is installed having the following features:
concode= 393216 tokenbase= 0
NAMELIST
 flags= 0 format= 1073741827 length= 0 maxlength= 0
text=
 valuecount= 0 rangecount= 0
FLAGLIST
 flags= 0 format= 1073741827 length= 24 maxlength= 0
text= , , , , , , , , , ,
 valuecount= 11 rangecount= 0
```

Example 2.20

This example shows the result of execution for a lot of methods, properties, passing through all LSX LC classes. To understand it, you should have aside, the print out of the example and to follow the code lines.
In order to achieve the objective of this example, do the following step:

Step A - 2.20

Create the following LotusScript code for BUTTON19:

```
Sub Click(Source As Button)
	Dim LC_S As New LCSession
	Dim SrcCon As New LCConnection("db2")
	Dim fldlst As New lcfieldlist
	Dim inclc As New lcfield(LCTYPE_INT)
	Dim i As Long
	Dim index As Long
	Dim dtype As Long
	Dim flags As Long
	Dim fname As String
	Dim fmsg As String
	Dim lcfield As lcfield
	Dim ref As lcfield
	On Error Goto ErrorHandler
	SrcCon.Userid="Administrator"
	SrcCon.Password="rac4you"
	SrcCon.Database="SAMPLE"
	SrcCon.MetaData="EMPLOYEE"
	srcCon.Disconnect
	LC_S.ClearStatus
	srcCon.Connect
	If(srcCon.select(Nothing,1,fldlst)=0) Then
		Messagebox "The MetaData table wasn't found"
		End
	End If
	Messagebox "The MetaData " & SrcCon.MetaData & " table was found"
	fmsg="There are " & fldlst.FieldCount & " columns in the " & SrcCon.MetaData & " table as follows:" & Chr(10)
	For i=1 To fldlst.FieldCount
		fmsg=fmsg & fldlst.getname(i) & Chr(10)
		REM Numele unei coloane se poate obtine si cu comanda fldlst.names(i-1) precum in linia de mai jos:
		REM fmsg=fmsg & fldlst.names(i-1) & Chr(10)
		If (fldlst.getname(i)="LASTNAME") Then
			Call fldlst.setname(i,"NLASTNAME")

REM The name of a field in fldlst can be as well changed using REPLACE method as in the below line:
REM Call fldlst.replace(i,"NLASTNAME",LCTYPE_TEXT)

		End If
	Next
	Messagebox fmsg
	Call fldlst.remove(9)
	fmsg="There are " & fldlst.FieldCount & " columns in the " & SrcCon.MetaData & " table as follows:" & Chr(10)
	For i=1 To fldlst.FieldCount
		fmsg=fmsg & fldlst.getname(i) & Chr(10)
	Next
	Messagebox fmsg
```

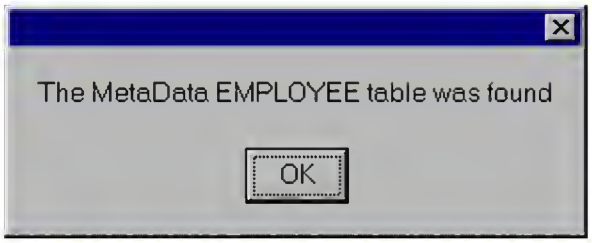
```
          Set lcfield=fldlst.insert(6,"COUNTRY",LCTYPE_TEXT)
          fmsg="There are " & fldlst.FieldCount & " columns in the " & SrcCon.MetaData & " table as follows:" & Chr(10)
          For i=1 To fldlst.FieldCount
                    fmsg=fmsg & fldlst.getname(i) & Chr(10)
          Next
          Messagebox fmsg
          fmsg="The " & SrcCon.MetaData & " Table Description is as follows:" & Chr(10)
          i=LCLIST_FIRST
          While (fldlst.list(i,,index,dtype,flags,fname)=True)
                    fmsg=fmsg & " index= " & index & " dtype= " & Cstr(dtype) & " flags= " & Hex(flags) & " fname= "
& fname & Chr(10)
                    i=LCLIST_NEXT
          Wend
          Messagebox fmsg
          inclc.flags=0
          Call fldlst.includefield(9,inclc,"CONTINENT")
          fmsg="The " & SrcCon.MetaData & " Table Description is as follows:" & Chr(10)
          i=LCL1ST_F1RST
          While (fldlst.list(i,,index,dtype,flags,fname)=True)
                    fmsg=fmsg & " index= " & index & " dtype= " & Cstr(dtype) & " flags= " & Hex(flags) & " fname= " &
fname & Chr(10)
                    i=LCL1ST_NEXT
          Wend
          Messagebox fmsg
          Set lcfield=fldlst.lookup("MIDINIT",i)
          1f Not(lcfield 1s Nothing) Then
                    Messagebox  "Found M1D1N1T in the fldlst at position " & i
                    Set ref=fldlst.copyfield(11,lcfield,"REGION")
                    fmsg="The " & SrcCon.MetaData & " Table Description is as follows:" & Chr(10)
                    i=LCL1ST_F1RST
                    While (fldlst.list(i,,index,dtype,flags,fname)=True)
                              fmsg=fmsg & " index= " & index & " dtype= " & Cstr(dtype) & " flags= " & Hex(flags) & "
fname= " & fname & Chr(10)
                              i=LCL1ST_NEXT
                    Wend
                    Messagebox fmsg
          Else
                    Messagebox "Didn't find M1D1N1T in the fldlst"
          End 1f
          srcCon.disconnect
          End
ErrorHandler:
          Messagebox "Attention ! You are in Error"
          Dim msg As String
          Dim errortext As String
          Dim msgcode As Long
          Dim status As Long
          1f (LC_S.status <> LCSUCCESS) Then
                    status=LC_S.getstatus(errortext,msgcode,msg)
                    Messagebox "1nternal Error Text= " & errortext & Chr(10) & "1nternal Error Code= " & status & Chr(10) _
                    & "External Error Text= " & msg & Chr(10) & "External Error Code= " & msgcode
          Else
                    Messagebox "Lotus Notes Error Text= " & Error() & Chr(10) & "Lotus Notes Error Code= " & Err()
          End 1f
          End
End Sub
```

In order to run **Example 2.20** do the following steps:

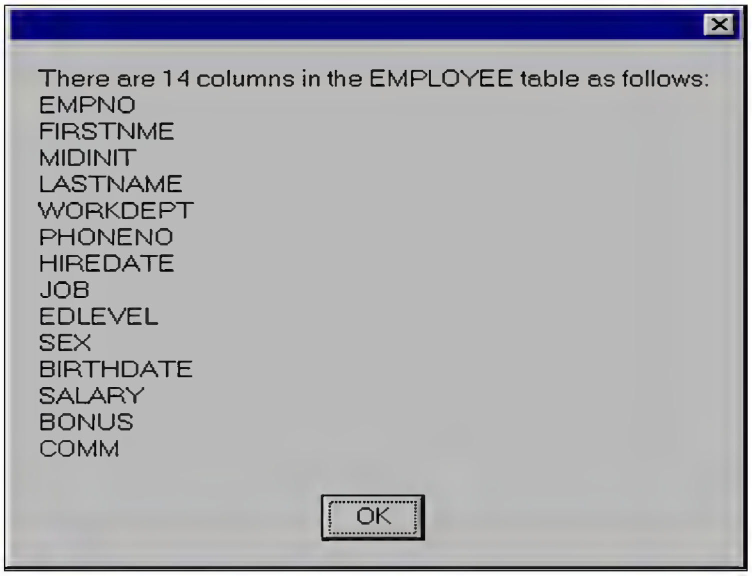
✓ Open the document created in Example 2.1, or create a new one; in both situations, when the exercise is done, you don't need to save the opened document.

✓ Push onto BUTTON19

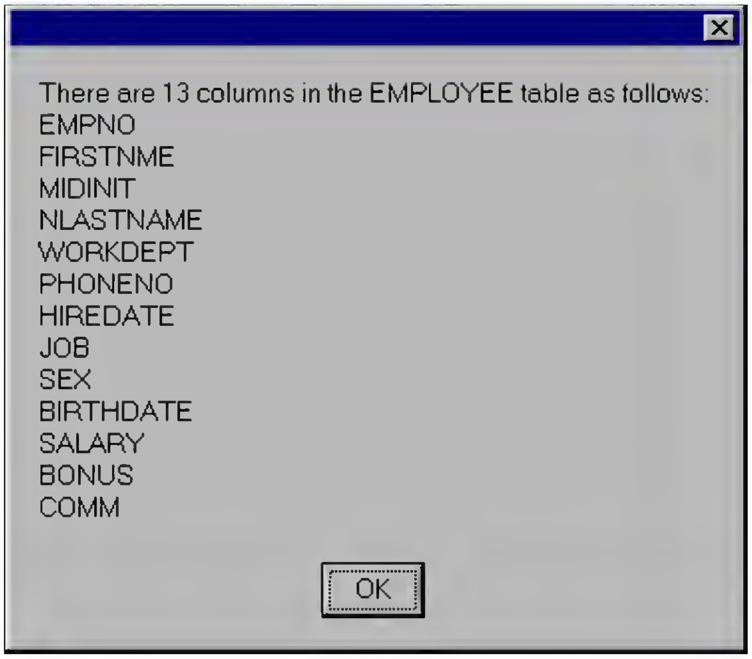
The result is as follows:



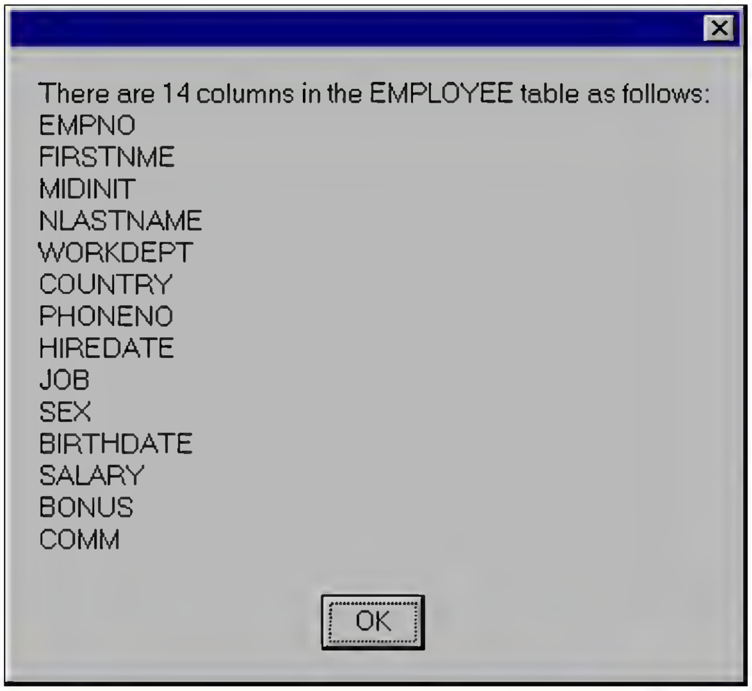
```
The MetaData EMPLOYEE table was found
```



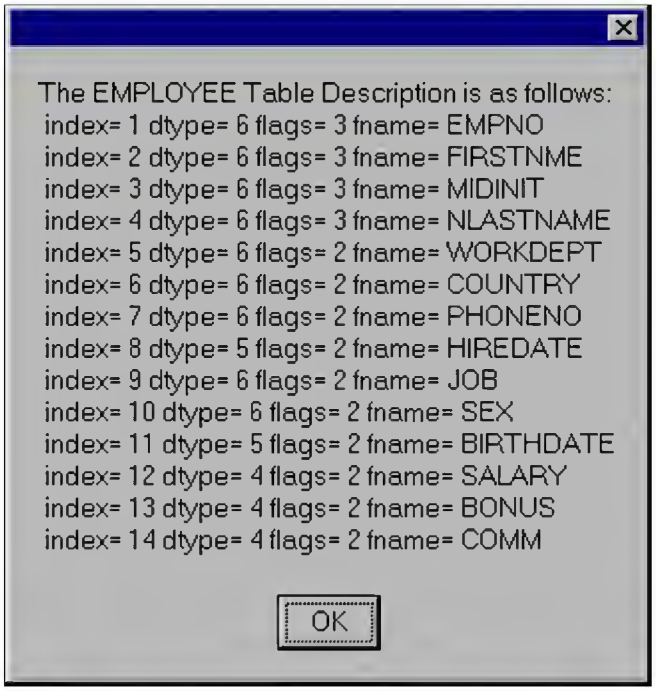
```
There are 14 columns in the EMPLOYEE table as follows:
EMPNO
FIRSTNME
MIDINIT
LASTNAME
WORKDEPT
PHONENO
HIREDATE
JOB
EDLEVEL
SEX
BIRTHDATE
SALARY
BONUS
COMM
```



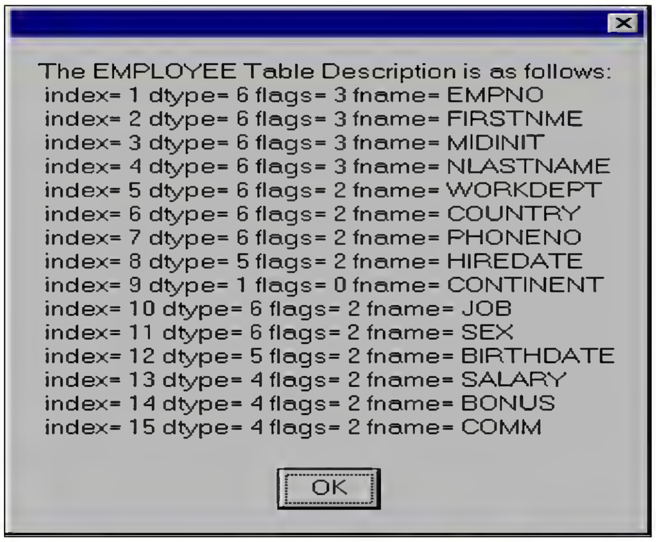
```
There are 13 columns in the EMPLOYEE table as follows:
EMPNO
FIRSTNME
MIDINIT
NLASTNAME
WORKDEPT
PHONENO
HIREDATE
JOB
SEX
BIRTHDATE
SALARY
BONUS
COMM
```



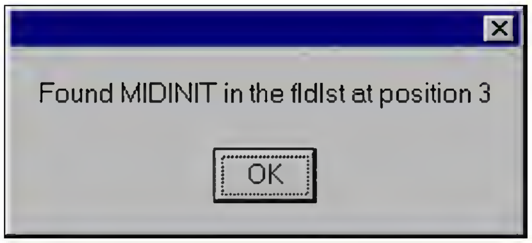
```
There are 14 columns in the EMPLOYEE table as follows:
EMPNO
FIRSTNME
MIDINIT
NLASTNAME
WORKDEPT
COUNTRY
PHONENO
HIREDATE
JOB
SEX
BIRTHDATE
SALARY
BONUS
COMM
```



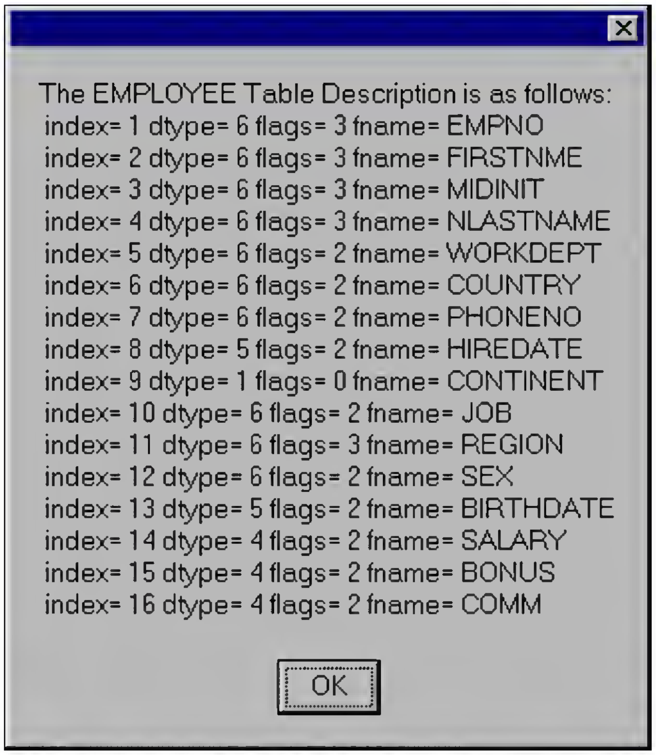
```
The EMPLOYEE Table Description is as follows:
 index= 1 dtype= 6 flags= 3 fname= EMPNO
 index= 2 dtype= 6 flags= 3 fname= FIRSTNME
 index= 3 dtype= 6 flags= 3 fname= MIDINIT
 index= 4 dtype= 6 flags= 3 fname= NLASTNAME
 index= 5 dtype= 6 flags= 2 fname= WORKDEPT
 index= 6 dtype= 6 flags= 2 fname= COUNTRY
 index= 7 dtype= 6 flags= 2 fname= PHONENO
 index= 8 dtype= 5 flags= 2 fname= HIREDATE
 index= 9 dtype= 6 flags= 2 fname= JOB
 index= 10 dtype= 6 flags= 2 fname= SEX
 index= 11 dtype= 5 flags= 2 fname= BIRTHDATE
 index= 12 dtype= 4 flags= 2 fname= SALARY
 index= 13 dtype= 4 flags= 2 fname= BONUS
 index= 14 dtype= 4 flags= 2 fname= COMM
```



```
The EMPLOYEE Table Description is as follows:
 index= 1 dtype= 6 flags= 3 fname= EMPNO
 index= 2 dtype= 6 flags= 3 fname= FIRSTNME
 index= 3 dtype= 6 flags= 3 fname= MIDINIT
 index= 4 dtype= 6 flags= 3 fname= NLASTNAME
 index= 5 dtype= 6 flags= 2 fname= WORKDEPT
 index= 6 dtype= 6 flags= 2 fname= COUNTRY
 index= 7 dtype= 6 flags= 2 fname= PHONENO
 index= 8 dtype= 5 flags= 2 fname= HIREDATE
 index= 9 dtype= 1 flags= 0 fname= CONTINENT
 index= 10 dtype= 6 flags= 2 fname= JOB
 index= 11 dtype= 6 flags= 2 fname= SEX
 index= 12 dtype= 5 flags= 2 fname= BIRTHDATE
 index= 13 dtype= 4 flags= 2 fname= SALARY
 index= 14 dtype= 4 flags= 2 fname= BONUS
 index= 15 dtype= 4 flags= 2 fname= COMM
```



```
Found MIDINIT in the fldlst at position 3
```



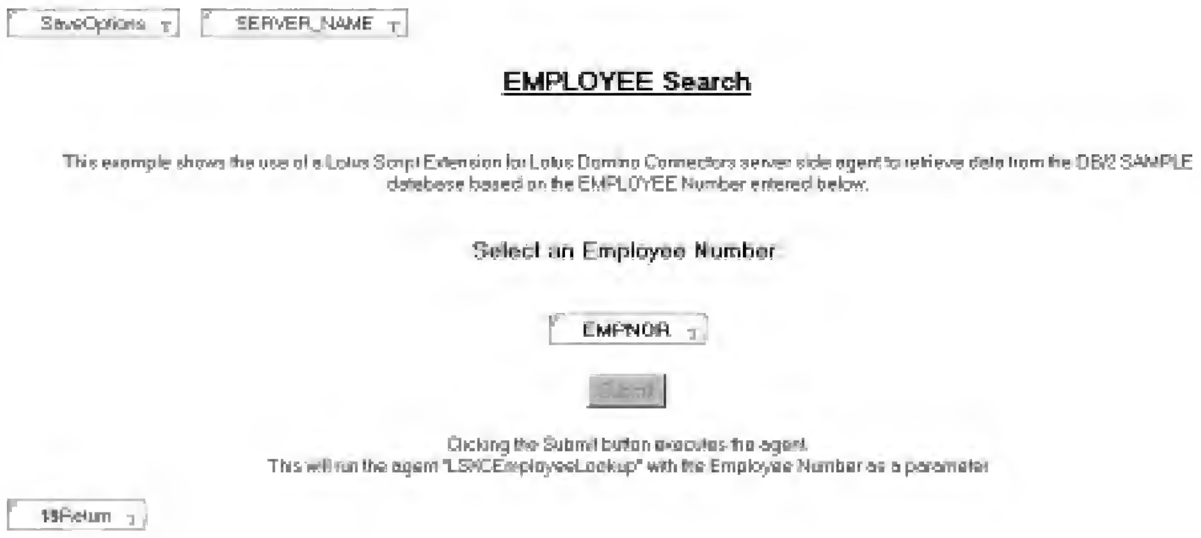
```
The EMPLOYEE Table Description is as follows:
 index= 1 dtype= 6 flags= 3 fname= EMPNO
 index= 2 dtype= 6 flags= 3 fname= FIRSTNME
 index= 3 dtype= 6 flags= 3 fname= MIDINIT
 index= 4 dtype= 6 flags= 3 fname= NLASTNAME
 index= 5 dtype= 6 flags= 2 fname= WORKDEPT
 index= 6 dtype= 6 flags= 2 fname= COUNTRY
 index= 7 dtype= 6 flags= 2 fname= PHONENO
 index= 8 dtype= 5 flags= 2 fname= HIREDATE
 index= 9 dtype= 1 flags= 0 fname= CONTINENT
 index= 10 dtype= 6 flags= 2 fname= JOB
 index= 11 dtype= 6 flags= 3 fname= REGION
 index= 12 dtype= 6 flags= 2 fname= SEX
 index= 13 dtype= 5 flags= 2 fname= BIRTHDATE
 index= 14 dtype= 4 flags= 2 fname= SALARY
 index= 15 dtype= 4 flags= 2 fname= BONUS
 index= 16 dtype= 4 flags= 2 fname= COMM
```

Example 2.21

This example shows how to access external databases via a Web browser and Domino Server, using LSX LC. To access the data from the Web browser, you must define a LSX LC connection to external data source and must write the LSX LC code in an agent that runs via a URL command. The display of the data needed to be formatted in HTML. In this example, giving the employee's serial number, we get information about an employee from SAMPLE database. **Example 2.21 is similar with Example 3.14; the only difference is that Example 2.21 uses LSX LC and Example 3.14 uses ODBC.**
In order to execute Example 2.21 do the following steps:

Step A - 2.21

Create a form on LSXCODBC.NSF, named FORM6 having the following structure:



SaveOptions SERVER_NAME

EMPLOYEE Search

This example shows the use of a Lotus Script Extension for Lotus Domino Connectors server side agent to retrieve data from the DB2 SAMPLE database based on the EMPLOYEE Number entered below.

Select an Employee Number

EMPNOR

Submit

Clicking the Submit button executes the agent.
This will run the agent "LSXCEmployeeLookup" with the Employee Number as a parameter

$$Return

Let's detail the above form:

- Field SaveOptions: text + computed, formula: "0"
- Field SERVER_NAME: text + computed, formula: SERVER_NAME
- Field EMPNOR: text + editable
- Field $$Return: text + computed, formula:
  @Return("[http://"+SERVER_NAME+"/"+@ReplaceSubstring(@Subset(@DbName;-1);"\\";"/")+"/LSX LCEmployeeLookup?OpenAgent&"+EMPNOR+"]")
- Button Submit: JavaScript Formula: this.form.submit()

*

* *

The Fields: SaveOptions, SERVER_NAME, $$Return have in "Paragraph Hide When":
Hide paragraph from:

* Notes R4.6 or later		* Web browser
Hide paragraph when document is:
* opened for reading		* opened for editing

* printed

Step B- 2.21

Create the agent named **LSX LCEmployeeLookup** having the features: Share Agent + Manually from agent list + Should act on all documents in database.

Create the following LotusScript code for agent **LSX LCEmployeeLookup**:

```
Option Public
Uselsx lc "*LSXLC"

Sub Initialize
        Dim lcs As New lcsession
        Dim lcfldlst As New lcfieldlist(1)
        Dim session As New  notessession
        Dim doc As notesdocument
        Dim conn As New lcconnection("db2")
        Dim query As  String
        Dim var1 As Integer
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim data1 As Long
        Set doc=session.documentcontext
        Set db=session.currentdatabase
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE1"
        userid="Administrator"
        parola="rac4you"
        conn.database=dsn
        conn.userid=userid
        conn.password=parola
        urlstring=doc.Query_String(0)
        urllength=Len(urlstring)
        paramposition=Instr(urlstring,"&")+1
        webparam=Mid(urlstring,paramposition,urllength-paramposition+1)
        conn.disconnect
        lcs.clearstatus
        On Error Goto et1
        conn.connect
        On Error Goto 0
        query="select * from EMPLOYEE where EMPNO='" & webparam & "'"
        On Error Goto et2
        data1=conn.execute(query,lcfldlst)
        On Error Goto 0
        If data1 <> 0 Then
                var1=0
                While (conn.fetch(lcfldlst) >0)
                        empno=lcfldlst.EMPNO(0)
                        If empno=webparam Then
```

```
                                        var1=1
                                        firstnme=lcfldlst.FIRSTNME(0)
                                        midinit=lcfldlst.MIDINIT(0)
                                        lastname=lcfldlst.LASTNAME(0)
                                        workdept=lcfldlst.WORKDEPT(0)
                                        phoneno=lcfldlst.PHONENO(0)
                                        hiredate=lcfldlst.HIREDATE(0)
                                        job=lcfldlst.JOB(0)
                                        edlevel=lcfldlst.EDLEVEL(0)
                                        sex=lcfldlst.SEX(0)
                                        birthdate=lcfldlst.BIRTHDATE(0)
                                        salary=lcfldlst.SALARY(0)
                                        bonus=lcfldlst.BONUS(0)
                                        comm=lcfldlst.COMM(0)
                                        Print "<head><body>"
                                        Print "<h3>This is the information for employee: " & webparam & "</h3>"
                                        Print "EMPNO: " & empno & "<br>"
                                        Print "FIRSTNAME: " & firstnme & "<br>"
                                        Print "MIDINIT: " & midinit & "<br>"
                                        Print "LASTNAME: " & lastname & "<br>"
                                        Print "<br>"
                                        Print "WORKDEPT: <a href=./LSX LCDeptLookup?OpenAgent&" & workdept &
">" & workdept & "</a>" & "<br>"
                                        Print "PHONENO: " & phoneno & "<br>"
                                        Print "HIREDATE: " & hiredate & "<br>"
                                        Print "JOB: " & job & "<br>"
                                        Print "EDLEVEL: " & edlevel & "<br>"
                                        Print "SEX: " & sex & "<br>"
                                        Print "BIRTHDATE: " & birthdate & "<br>"
                                        Print "SALARY: " & salary & "<br>"
                                        Print "BONUS: " & bonus & "<br>"
                                        Print "COMM: " & comm & "<br>"
                                        Print "<br><br>"
                                        Print "Thank You"
                                End If
                        Wend
                        If var1 <> 1 Then
                                Goto et4
                        End If
                Else
                        Goto et4
                End If
                conn.disconnect
                End
et1:
                Print "Not OK, Could not connect to " & dsn & " DataBase.<br>"
                Goto et3
et2:
                conn.disconnect
                Print "Not OK, Could not Select from " & dsn & " DataBase !<br>"
                Goto et3
et4:
                conn.disconnect
                Print "Not OK, The EMPLOYEE ID cannot be found in " & dsn & " DataBase !<br>"
et3:
                If (lcs.status <> LCSUCCSESS) Then
                        data1=lcs.getstatus(errortext,msgcode,msg)
                        Print "Internal Error Text: " & errortext & "<br>"
                        Print "Internal Error Code: " & Str$(data1) & "<br>"
                        Print "External Error Text: " & msg & "<br>"
                        Print "External Error Code: " & Str$(msgcode) & "<br>"
```

```
        Else
                Print "Lotus Notes Text Error: " & Error() & "<br>"
                Print "Lotus Notes Code Error: " & Str$(Err()) & "<br>"
        End If
End Sub
```

Step C- 2.21

Create the agent named **LSX LCDeptLookup** having the features: Share Agent + Run once(@command may be used).

Create the following LotusScript code for agent **LSX LCDeptLookup**:

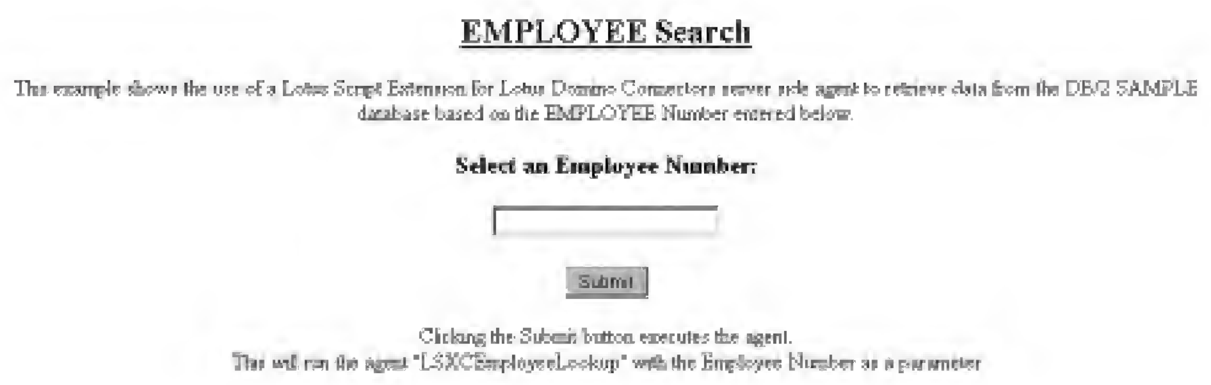
```
Option Public
Uselsx lc "*LSXLC"

Sub Initialize
        Dim lcs As New lcsession
        Dim lcfldlst As New lcfieldlist(1)
        Dim session As New  notessession
        Dim doc As notesdocument
        Dim conn As New lcconnection("db2")
        Dim query As  String
        Dim msg As String
        Dim errortext As String
        Dim msgcode As Long
        Dim data1 As Long
        Set doc=session.documentcontext
        Set db=session.currentdatabase
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        conn.database=dsn
        conn.userid=userid
        conn.password=parola
        urlstring=doc.Query_String(0)
        urllength=Len(urlstring)
        paramposition=Instr(urlstring,"&")+1
        webparam=Mid(urlstring,paramposition,urllength-paramposition+1)
        conn.disconnect
        lcs.clearstatus
        On Error Goto et1
        conn.connect
        On Error Goto 0
        query="select * from EMPLOYEE where WORKDEPT='" & webparam & "'"
        On Error Goto et2
        data1=conn.execute(query,lcfldlst)
        On Error Goto 0
        Print "<head><body>"
        Print "<h3>These are other employees that work in department " & webparam & "</h3>"
        Print "<table border="1">"
        Print "<tr>"
        Print "<td>EMPNO</td>"
        Print "<td>FIRSTNME</td>"
        Print "<td>MIDINIT</td>"
        Print "<td>LASTNAME</td>"
        Print "<td>PHONENO</td>"
        Print "<td>HIREDATE</td>"
        Print "<td>JOB</td>"
        Print "<td>EDLEVEL</td>"
        Print "<td>SEX</td>"
        Print "<td>BIRTHDATE</td>"
        Print "<td>SALARY</td>"
        Print "<td>BONUS</td>"
        Print "<td>COMM</td>"
        Print "<tr>"
        While (conn.fetch(lcfldlst)>0)
                empno=lcfldlst.EMPNO(0)
                firstnme=lcfldlst.FIRSTNME(0)
                midinit=lcfldlst.MIDINIT(0)
                lastname=lcfldlst.LASTNAME(0)
                phoneno=lcfldlst.PHONENO(0)
                hiredate=lcfldlst.HIREDATE(0)
                job=lcfldlst.JOB(0)
                edlevel=lcfldlst.EDLEVEL(0)
                sex=lcfldlst.SEX(0)
                birthdate=lcfldlst.BIRTHDATE(0)
                salary=lcfldlst.SALARY(0)
                bonus=lcfldlst.BONUS(0)
                comm=lcfldlst.COMM(0)
                Print "<tr>"
                Print "<td>" & empno & "</tr>"
                Print "<td><a href=./LSX LCEmployeeLookup?OpenAgent&" & empno & ">" & firstnme & "</a>"
& "</tr>"
                Print "<td>" & midinit & "</tr>"
                Print "<td>" & lastname & "</tr>"
                Print "<td>" & phoneno & "</tr>"
                Print "<td>" & hiredate & "</tr>"
                Print "<td>" & job & "</tr>"
                Print "<td>" & edlevel & "</tr>"
                Print "<td>" & sex & "</tr>"
                Print "<td>" & birthdate & "</tr>"
                Print "<td>" & salary & "</tr>"
                Print "<td>" & bonus & "</tr>"
                Print "<td>" & comm & "</tr>"
                Print "</tr>"
                Print "</br>"
        Wend
        Print "</table>"
        Print "</body></head>"
        conn.disconnect
        End
et1:
        Print "Not OK, Could not connect to " & dsn & " DataBase.<br>"
        Goto et3
et2:
        conn.disconnect
        Print "Not OK, Could not Select from " & dsn & " DataBase !<br>"
et3:
        If (lcs.status <> LCSUCCSESS) Then
                data1=lcs.getstatus(errortext,msgcode,msg)
                Print "Internal Error Text: " & errortext & "<br>"
                Print "Internal Error Code: " & Str$(data1) & "<br>"
                Print "External Error Text: " & msg & "<br>"
                Print "External Error Code: " & Str$(msgcode) & "<br>"
        Else
```

```
                Print "Lotus Notes Text Error: " & Error() & "<br>"
                Print "Lotus Notes Code Error: " & Str$(Err()) & "<br>"
        End If
End Sub
```

In order to run **Example 2.21** do the following steps:

✓ Open a Web browser and type the following URL:
http://mummer.ism.can.ibm.com/test1/lsxcodbc.nsf/form6

The result on the Web browser is as follows:



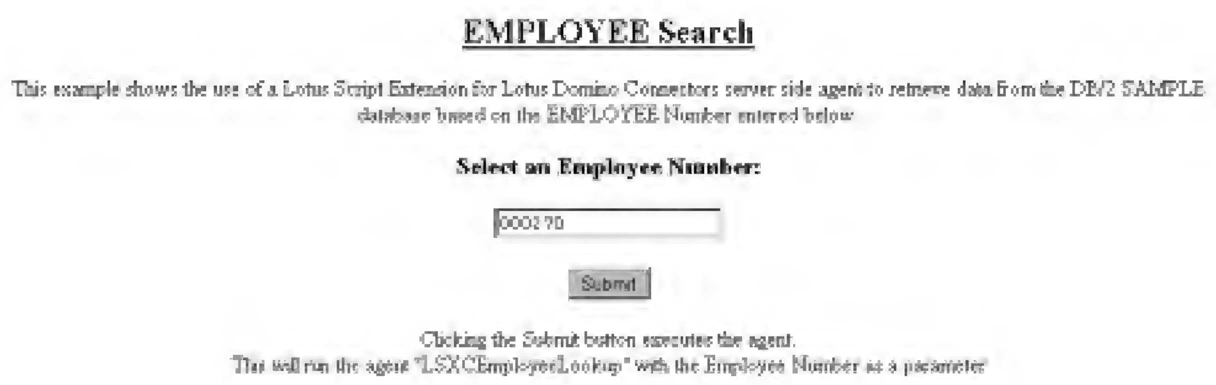
EMPLOYEE Search

This example shows the use of a Lotus Script Extension for Lotus Domino Connectors server side agent to retrieve data from the DB/2 SAMPLE database based on the EMPLOYEE Number entered below.

Select an Employee Number:

Clicking the Submit button executes the agent.
This will run the agent "LSXCEmployeeLookup" with the Employee Number as a parameter

✓ Type the following Serial Number:**000270** and Click onto **Submit** button when finished.



EMPLOYEE Search

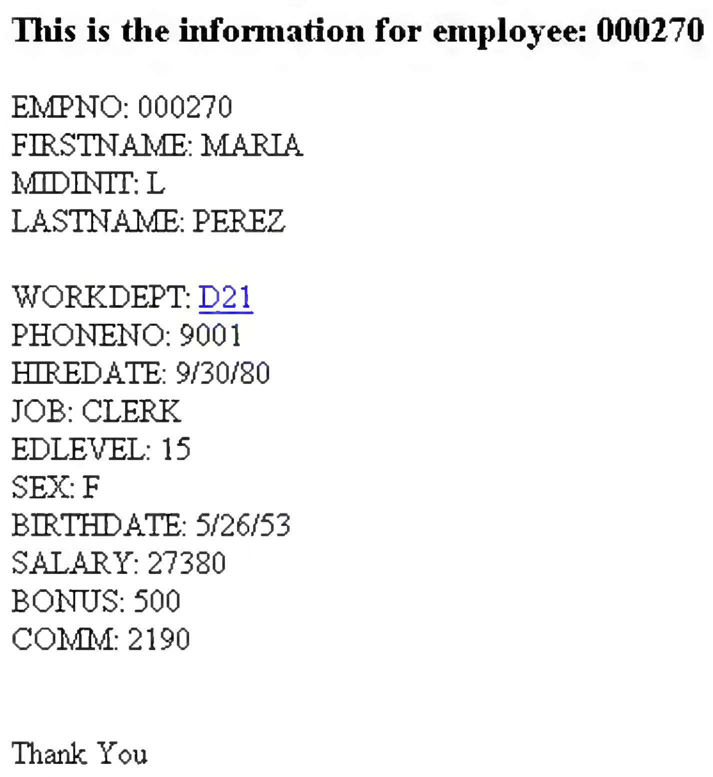
This example shows the use of a Lotus Script Extension for Lotus Domino Connectors server side agent to retrieve data from the DB/2 SAMPLE database based on the EMPLOYEE Number entered below.

Select an Employee Number:

000270

Clicking the Submit button executes the agent.
This will run the agent "LSXCEmployeeLookup" with the Employee Number as a parameter

After a while, the Web browser brings up the following information:



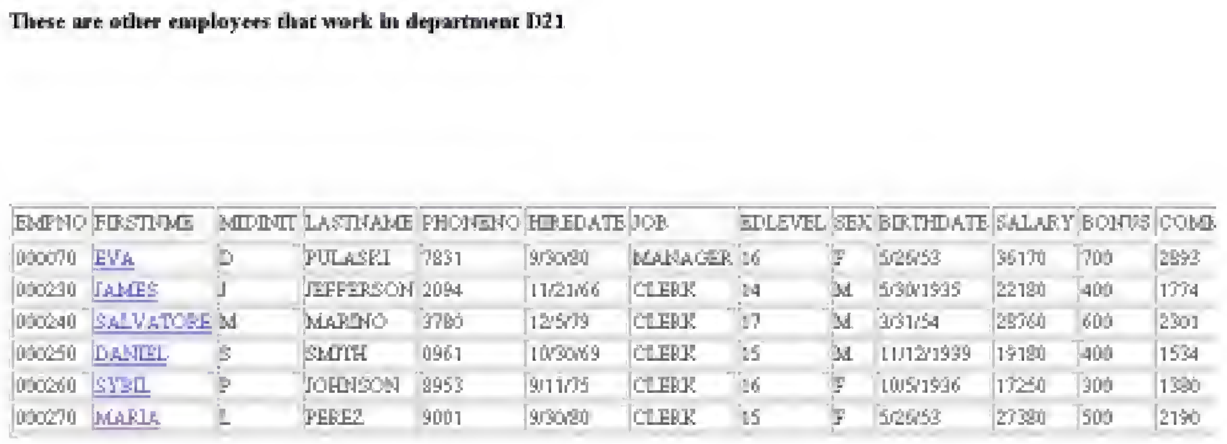
This is the information for employee: 000270

EMPNO: 000270
FIRSTNAME: MARIA
MIDINIT: L
LASTNAME: PEREZ

WORKDEPT: D21
PHONENO: 9001
HIREDATE: 9/30/80
JOB: CLERK
EDLEVEL: 15
SEX: F
BIRTHDATE: 5/26/53
SALARY: 27380
BONUS: 500
COMM: 2190

Thank You

✓ Click on **D21** Reference Link in order to see what other persons work in the same Department.



These are other employees that work in department D21

EMPNO	FIRSTNME	MIDINIT	LASTNAME	PHONENO	HIREDATE	JOB	EDLEVEL	SEX	BIRTHDATE	SALARY	BONUS	COMM
000070	EVA	D	PULASKI	7831	9/30/80	MANAGER	16	F	5/26/53	36170	700	2892
000230	JAMES	J	JEFFERSON	2094	11/21/66	CLERK	14	M	5/30/1935	22180	400	1774
000240	SALVATORE	M	MARINO	3780	12/5/79	CLERK	17	M	3/31/54	28760	600	2301
000250	DANIEL	S	SMITH	0961	10/30/69	CLERK	15	M	11/12/1939	19180	400	1534
000260	SYBIL	P	JOHNSON	8953	9/11/75	CLERK	16	F	10/5/1936	17250	300	1380
000270	MARIA	L	PEREZ	9001	9/30/80	CLERK	15	F	5/26/53	27380	500	2190

✓ Click on any Name, listed under column FIRSTNME. Actually behind each name is a Reference Link. After a while the Web browser brings up the information for that specific Name in the same format as for **Maria Perez**: **This is the information for employee .....**

You can play around selecting a lot of EMPNOs and FIRSTNMEs from EMPLOYEE table.

Let’s try to explain what happened.

After the user click the Submit button from the browser, the formula of the $$Return field is evaluated and then executed on Domino Server.
When the user selects an employee number, the $$Return field is evaluated to the following URL, which is processed by the Domino Web Server and runs the LSX LCEmployeeLookup agent with a parameter of 000270:

http://mummer.ism.can.ibm.com/test1/lsxcodbc.nsf/LSX LCEmployeeLookup?OpenAgent&000270

As the agent is initiated, it parses the command line that was passed to it "via" the Domino Context method of the NotesSession class, which at it turn gives us access to the CGI variable URL String. The following code shows this:

```
Set doc=session.documentcontext
Set db=session.currentdatabase
conn.silentmode=True
Dim dsn As String
Dim userid As String
Dim parola As String
dsn="SAMPLE"
userid="Administrator"
parola="rac4you"
urlstring=doc.Query_String(0)
urllength=Len(urlstring)
paramposition=Instr(urlstring,"&")+1
webparam=Mid(urlstring,paramposition,urllength-paramposition+1)
```

Knowing the employee number, we can use LSX LC to query the EMPLOYEE table in the SAMPLE database and pull the information for EMPNO=000270 as follows:

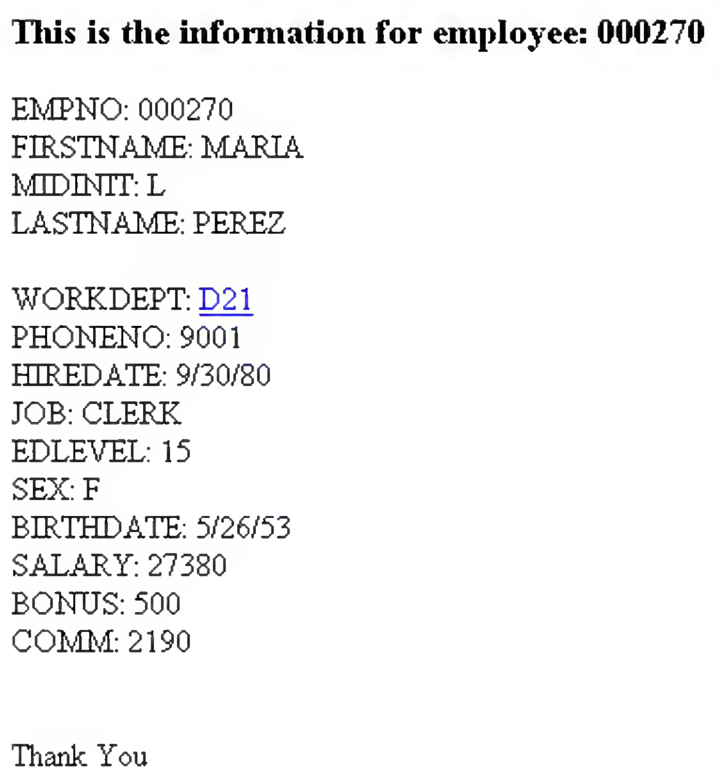
```
query.sql="select * from EMPLOYEE where EMPNO='" & webparam & "'"
```

The final action is to display the information onto the Web browser using LotusScript Print command and a combination of HTML tags.

But what about displaying other employees from the same department ?
For this, take a look at the following code line in LSX LCEmployeeLookup agent:

```
                    Print "WORKDEPT: <a href=./LSX LCDeptLookup?OpenAgent&" & workdept & ">" &
workdept & "</a>" & "<br>"
```

This line create an HTML link to another agent called LSX LCDeptLookup as it's shown below:



This is the information for employee: 000270

EMPNO: 000270
FIRSTNAME: MARIA
MIDINIT: L
LASTNAME: PEREZ

WORKDEPT: D21
PHONENO: 9001
HIREDATE: 9/30/80
JOB: CLERK
EDLEVEL: 15
SEX: F
BIRTHDATE: 5/26/53
SALARY: 27380
BONUS: 500
COMM: 2190

Thank You

Clicking the URL aside of word **WORKDEPT** that means, the word **D21** , will run the agent LSX LCDeptLookup on the Domino Server with a parameter of D21. This agent retrieves a list of all employee that work in the same department and displays the information onto the Web browser using LotusScript Print command and a combination of HTML tags.

Having the table generated by LSX LCDeptLookup agent, we can click onto any names shown in order to get information for a specific employee; in reality, we invoke again the LSX LCEmployeeLookup agent with the following code line:

```
                    Print "<td><a href=./LSX LCEmployeeLookup?OpenAgent&" & empno & ">" & firstnme & "</a>"
& "</tr>"
```

3. LotusScript Data Object(LS:DO)

All examples in this chapter deal with LS:DO, using the same database(LSXCODBC.NSF) and Domino Server(MUMMER.ISM.CAN.IBM.COM) that have been defined in Chapter 2(LotusScript Extension for Lotus Domino Connectors)

When you decide to study the examples of this Chapter, you should have aside the following books:

- Lotus Domino Release 5.0: A Developer's Handbook(IBM Redbook SG24-5331-01)
- Domino Release 5. Domino Designer Programming Guide, Volume 2:LotusScript Classes

For a program that deals with ODBC, the following statements should be executed:

USELSX "*LSXODBC"

- Declare a new object of ODBCConnection type (dim con as new odbcconnection)
- Declare a new object of ODBCQuery type (dim qry as new odbcquery)
- Declare a new object of ODBCResultSet type (dim result as new odbcresult)
- Connect to a DataBaseSource type (con.connectto(.......))
- Associate the object of ODBCConnection type to the object of ODBCQuery type(set qry.connection=con)
- Associate the object of ODBCResultSet to the object of ODBCQuery(set result.query=qry)
- Specify a query(qry.SQL=".................")
- Execute a query(result.execute)
- Examine ResultSet.

Example 3.1

This example displays the name of the available data sources
In order to achieve this objective do the following steps:

Step A - 3.1

Create the agent AGENT1 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.1

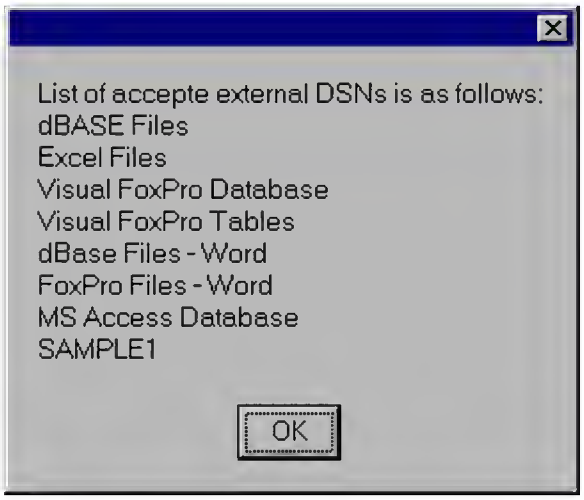
Create the following LotusScript code for AGENT1:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim msg As String
        Dim dsnlist As Variant
        dsnlist=con.listdatasources
        For n%=Lbound(dsnlist) To Ubound(dsnlist)
                msg=msg & dsnlist(n%) & Chr(10)
        Next
        Messagebox "List of accepted external DSNs is as follows:" & Chr(10) & msg
End Sub
```

In order to run **AGENT1** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT1

The result is as follows:



```
List of accepte external DSNs is as follows:
dBASE Files
Excel Files
Visual FoxPro Database
Visual FoxPro Tables
dBase Files - Word
FoxPro Files - Word
MS Access Database
SAMPLE1
```

Example 3.2

This example shows an agent connection to the data source. If the connection fails the agent exits, contrary the agent lists the tables for the data source, looping through a string array returned by ListTables.
In order to achieve this objective do the following steps:

Step A - 3.2

Create the agent AGENT2 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.2

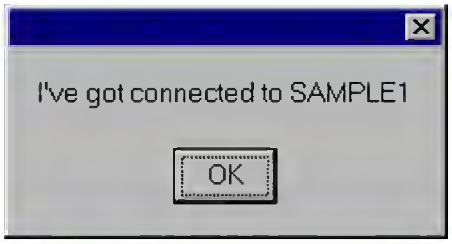
Create the following LotusScript code for AGENT2:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        Dim  msg As String
        Dim tables As Variant
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Call con.disconnect
        Call con.connectto(dsn, userid, parola)
        If Not con.isconnected Then
                Messagebox "I cannot get connected to " & dsn
                End
        End If
        Messagebox "I've got connected to " & dsn
        con.silentmode=False
        tables=con.listtables(dsn,userid,parola)
        msg="An array has been created having minimum " & Lbound(tables) & " and maximum " & Ubound(tables) & "
tables:" & Chr(10)
        For n%=Lbound(tables) To Ubound(tables)
                msg=msg & tables(n%) & " , "
        Next
        Messagebox msg
        con.disconnect
End Sub
```

In order to run **AGENT2** do the following step:

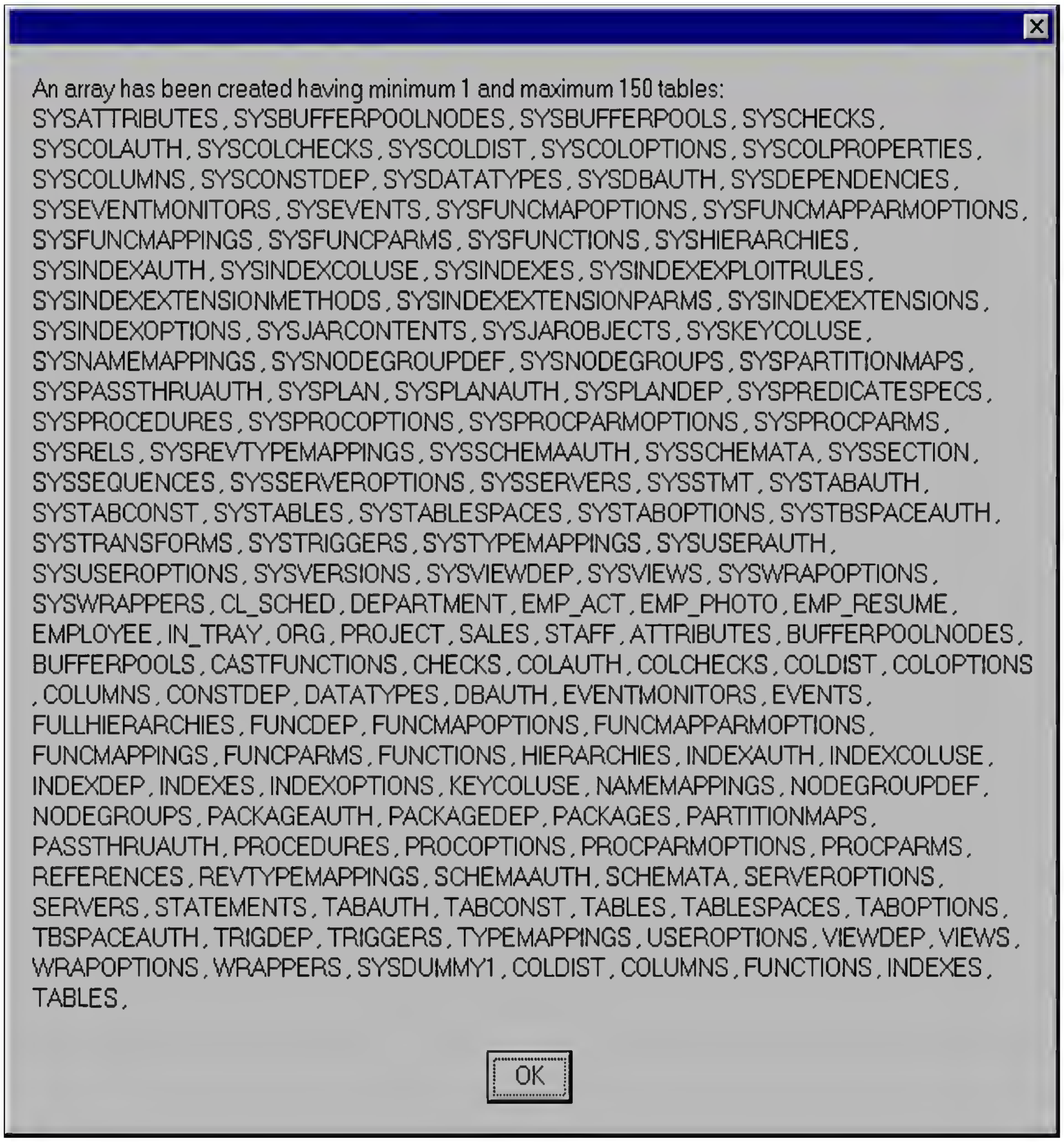
✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT2

The result is as follows:



```
I've got connected to SAMPLE1
```

OK



```
An array has been created having minimum 1 and maximum 150 tables:
SYSATTRIBUTES , SYSBUFFERPOOLNODES , SYSBUFFERPOOLS , SYSCHECKS ,
SYSCOLAUTH , SYSCOLCHECKS , SYSCOLDIST , SYSCOLOPTIONS , SYSCOLPROPERTIES ,
SYSCOLUMNS , SYSCONSTDEP , SYSDATATYPES , SYSDBAUTH , SYSDEPENDENCIES ,
SYSEVENTMONITORS , SYSEVENTS , SYSFUNCMAPOPTIONS , SYSFUNCMAPPARMOPTIONS ,
SYSFUNCMAPPINGS , SYSFUNCPARMS , SYSFUNCTIONS , SYSHIERARCHIES ,
SYSINDEXAUTH , SYSINDEXCOLUSE , SYSINDEXES , SYSINDEXEXPLOITRULES ,
SYSINDEXEXTENSIONMETHODS , SYSINDEXEXTENSIONPARMS , SYSINDEXEXTENSIONS ,
SYSINDEXOPTIONS , SYSJARCONTENTS , SYSJAROBJECTS , SYSKEYCOLUSE ,
SYSNAMEMAPPINGS , SYSNODEGROUPDEF , SYSNODEGROUPS , SYSPARTITIONMAPS ,
SYSPASSTHRUAUTH , SYSPLAN , SYSPLANAUTH , SYSPLANDEP , SYSPREDICATESPECS ,
SYSPROCEDURES , SYSPROCOPTIONS , SYSPROCPARMOPTIONS , SYSPROCPARMS ,
SYSRELS , SYSREVTYPEMAPPINGS , SYSSCHEMAAUTH , SYSSCHEMATA , SYSSECTION ,
SYSSEQUENCES , SYSSERVEROPTIONS , SYSSERVERS , SYSSTMT , SYSTABAUTH ,
SYSTABCONST , SYSTABLES , SYSTABLESPACES , SYSTABOPTIONS , SYSTBSPACEAUTH ,
SYSTRANSFORMS , SYSTRIGGERS , SYSTYPEMAPPINGS , SYSUSERAUTH ,
SYSUSEROPTIONS , SYSVERSIONS , SYSVIEWDEP , SYSVIEWS , SYSWRAPOPTIONS ,
SYSWRAPPERS , CL_SCHED , DEPARTMENT , EMP_ACT , EMP_PHOTO , EMP_RESUME ,
EMPLOYEE , IN_TRAY , ORG , PROJECT , SALES , STAFF , ATTRIBUTES , BUFFERPOOLNODES ,
BUFFERPOOLS , CASTFUNCTIONS , CHECKS , COLAUTH , COLCHECKS , COLDIST , COLOPTIONS
, COLUMNS , CONSTDEP , DATATYPES , DBAUTH , EVENTMONITORS , EVENTS ,
FULLHIERARCHIES , FUNCDEP , FUNCMAPOPTIONS , FUNCMAPPARMOPTIONS ,
FUNCMAPPINGS , FUNCPARMS , FUNCTIONS , HIERARCHIES , INDEXAUTH , INDEXCOLUSE ,
INDEXDEP , INDEXES , INDEXOPTIONS , KEYCOLUSE , NAMEMAPPINGS , NODEGROUPDEF ,
NODEGROUPS , PACKAGEAUTH , PACKAGEDEP , PACKAGES , PARTITIONMAPS ,
PASSTHRUAUTH , PROCEDURES , PROCOPTIONS , PROCPARMOPTIONS , PROCPARMS ,
REFERENCES , REVTYPEMAPPINGS , SCHEMAAUTH , SCHEMATA , SERVEROPTIONS ,
SERVERS , STATEMENTS , TABAUTH , TABCONST , TABLES , TABLESPACES , TABOPTIONS ,
TBSPACEAUTH , TRIGDEP , TRIGGERS , TYPEMAPPINGS , USEROPTIONS , VIEWDEP , VIEWS ,
WRAPOPTIONS , WRAPPERS , SYSDUMMY1 , COLDIST , COLUMNS , FUNCTIONS , INDEXES ,
TABLES ,
```

OK

Example 3.3

This example passes through all rows of EMPLOYEE table and gets FIRSTNME and LASTNAME found in each row.
In order to achieve this objective do the following steps:

Step A - 3.3

Create the agent AGENT3 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.3

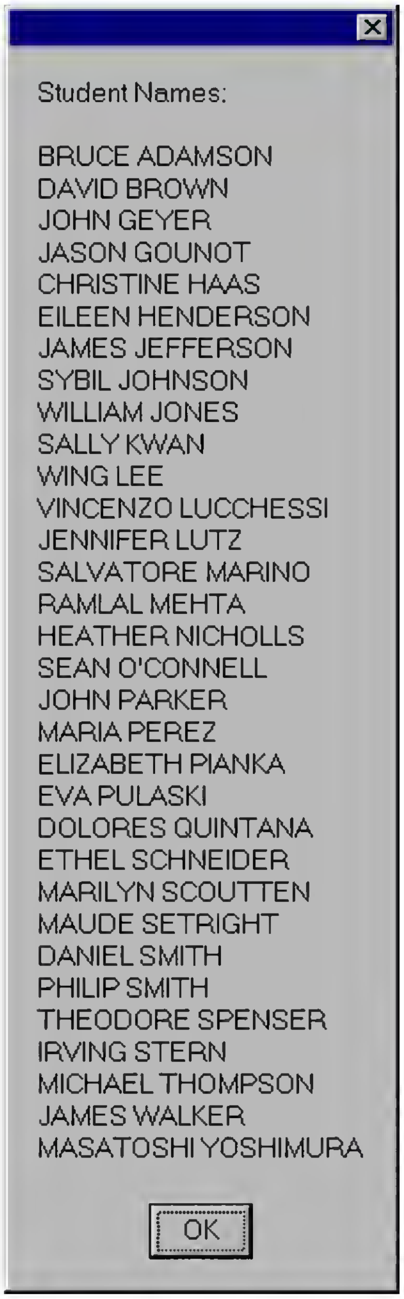
Create the following LotusScript code for AGENT3:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim firstnme As String
        Dim lastname As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        Dim  msg As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Call con.disconnect
        If Not con.connectto(dsn,userid,parola) Then
                Messagebox "Could not connect to " & dsn & " DataBase"
                End
        End If
        Set qry.connection=con
        Set result.query=qry
        qry.SQL="SELECT * FROM EMPLOYEE ORDER BY LASTNAME"
        result.execute
        msg="Student Names:" & Chr(10)
        If result.isresultsetavailable Then
                Do
                        result.nextrow
                        firstnme=result.getvalue("FIRSTNME")
                        lastname=result.getvalue("LASTNAME")
                        msg=msg & Chr(10) & firstnme & " " & lastname
                Loop Until result.isendofdata
                result.close(DB_CLOSE)
        Else
                Messagebox "No Data retrieved for EMPLOYEE Table"
                con.disconnect
                End
        End If
        Messagebox msg
        con.disconnect
End Sub
```

In order to run **AGENT3** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT3

The result is as follows:



```
Student Names:

BRUCE ADAMSON
DAVID BROWN
JOHN GEYER
JASON GOUNOT
CHRISTINE HAAS
EILEEN HENDERSON
JAMES JEFFERSON
SYBIL JOHNSON
WILLIAM JONES
SALLY KWAN
WING LEE
VINCENZO LUCCHESSI
JENNIFER LUTZ
SALVATORE MARINO
RAMLAL MEHTA
HEATHER NICHOLLS
SEAN O'CONNELL
JOHN PARKER
MARIA PEREZ
ELIZABETH PIANKA
EVA PULASKI
DOLORES QUINTANA
ETHEL SCHNEIDER
MARILYN SCOUTTEN
MAUDE SETRIGHT
DANIEL SMITH
PHILIP SMITH
THEODORE SPENSER
IRVING STERN
MICHAEL THOMPSON
JAMES WALKER
MASATOSHI YOSHIMURA
```

Example 3.4

This example sets the parameters in an SQL query then using NumParameters as upper bound, makes a loop in order to retrieve the row containing FIRSTNME and LASTNAME. In order to achieve this objective do the following steps:

Step A - 3.4

Create the agent AGENT4 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.4

Create the following LotusScript code for AGENT4:

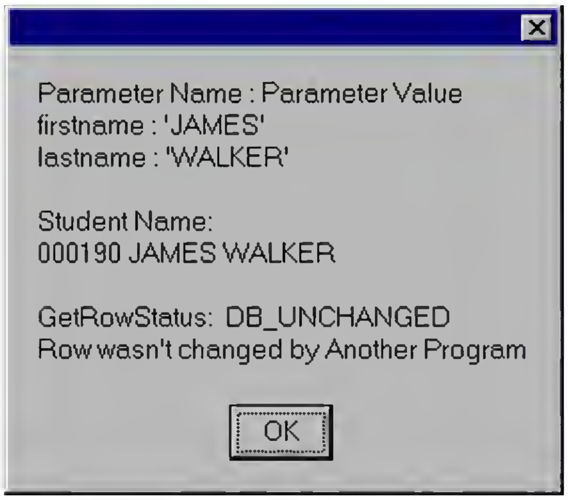
```
Sub Initialize
	Dim con As New odbcconnection
	Dim qry As New odbcquery
	Dim result As New odbcresultset
	Dim inputparameter As String
	Dim firstname As String
	Dim lastname As String
	Dim msg As String
	Dim dsn As String
	Dim userid As String
	Dim parola As String
	dsn="SAMPLE"
	userid="Administrator"
	parola="rac4you"
	Set qry.connection=con
	Set result.query=qry
	Call con.disconnect
	Call con.connectto(dsn,userid,parola)
	qry.sql="select * from EMPLOYEE where FIRSTNME= ?firstname? AND LASTNAME= ?lastname?"
	For i=1 To result.numparameters
		inputparameter=Inputbox$(result.getparametername(i),"Parameter" & i)
		Call result.setparameter(i, "'" & inputparameter & "'")
	Next
	msg="Parameter Name : Parameter Value"
	For i=1 To result.numparameters
		msg=msg & Chr(10) & result.getparametername(i) & " : " & result.getparameter(i)
	Next
	result.execute
	msg=msg & Chr(10) & Chr(10) & "Student Name:"
	If result.isresultsetavailable Then
		result.nextrow
		studentno=result.getvalue("EMPNO",studentno)
		firstname=result.getvalue("FIRSTNME",firstname)
		lastname=result.getvalue("LASTNAME",lastname)
		If result.isvaluealtered("EMPNO") Then
			msg=msg & Chr(10) & "The value in EMPNO field is altered" & Chr(10)
		End If
		If result.isvaluealtered("FIRSTNME") Then
			msg=msg & Chr(10) & "The value in FIRSTNME field is altered" & Chr(10)
		End If
```

```
                If result.isvaluealtered("LASTNAME") Then
                        msg=msg & Chr(10) & "The value in LASTNAME field is altered" & Chr(10)
                End If
                If result.isvaluenull("EMPNO") Then
                        msg=msg & Chr(10) & "The value in EMPNO field is NULL" & Chr(I0)
                End If
                If result.isvaluenull("FIRSTNME") Then
                        msg=msg & Chr(10) & "The value in FIRSTNME field is NULL" & Chr(10)
                End If
                If result.isvaluenull("LASTNAME") Then
                        msg=msg & Chr(10) & "The value in LASTNAME field is NULL" & Chr(10)
                End If
                msg=msg & Chr(10) & studentno & " " & firstname & " " & lastname
                msg=msg & Chr(10) & Chr(I0) & "GetRowStatus: "
                Select Case result.getrowstatus
                Case DB_UNCHANGED :  msg=msg & " DB_UNCHANGED"
                Case DB_ALTERED : msg=msg & " DB_ALTERED"
                Case DB_UPDATED : msg=msg & "DB_UPDATED"
                Case DB_DELETED : msg=msg & "DB_DELETED"
                Case DB_NEWROW : msg=msg & "DB_NEWROW"
                End Select
                If result.hasrowchanged Then
                        msg=msg & Chr(10) & "Another Program changed this row"
                Else
                        msg=msg & Chr(10) & "Row wasn't changed by Another Program"
                End If
        Else
                Messagebox "Cannot get result set"
                Call con.disconnect
                End
        End If
        Messagebox  msg
        Call con.disconnect
End Sub
```

In order to run **AGENT4** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT4

The result is as follows:



```
Parameter Name : Parameter Value
firstname : 'JAMES'
lastname : 'WALKER'

Student Name:
000190 JAMES WALKER

GetRowStatus:  DB_UNCHANGED
Row wasn't changed by Another Program
```

Example 3.5

This example examines all the fields (columns) in the EMPLOYEE table and displays their features
In order to achieve this objective do the following steps:

Step A - 3.5

Create the agent AGENT5 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.5

Create the following LotusScript code for AGENT5:

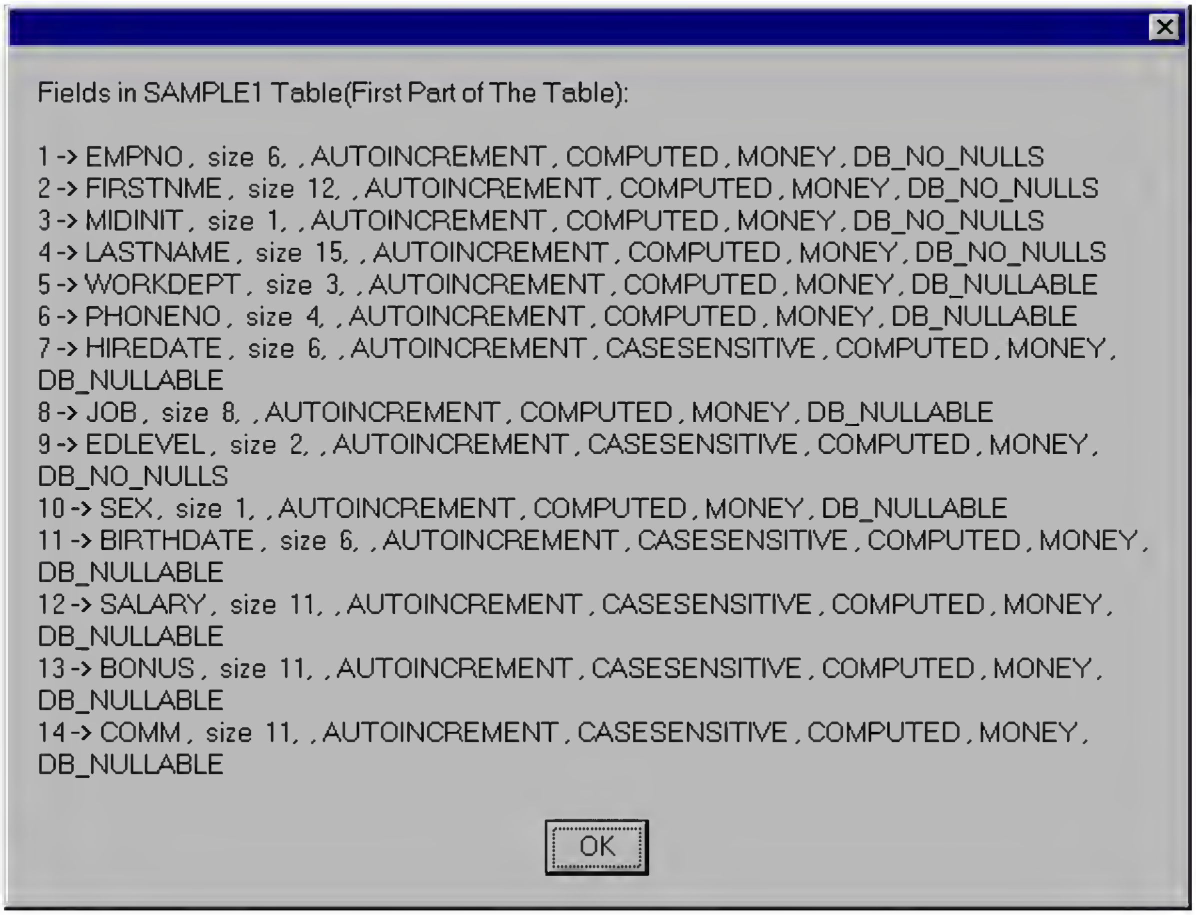
```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim fieldinfo As Variant
        Dim m2 As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                Messagebox "Could not connect to " & dsn & " DataBase"
                End
        End If
        qry.sql="select * from EMPLOYEE order by LASTNAME"
        result.execute
        If Not result.isresultsetavailable Then
                Messagebox "Couldn't get result set"
                Call con.disconnect
                End
        End If
        msg="Fields in " & dsn & " Table(First Part of The Table):" & Chr(10)
        For i=1 To result.numcolumns
                fieldinfo=result.fieldinfo(i)
                m2=""
                If fieldinfo(DB_INFO_AUTOINCREMENT)=DB_AUTOINCREMENT Then
                        m2=m2 & " , " & "AUTOINCREMENT"
                End If
                If fieldinfo(DB_INFO_CASESENSITIVE)=DB_CASESENSITIVE Then
                        m2=m2 & " , " & "CASESENSITIVE"
                End If
                If fieldinfo(DB_INFO_COMPUTED)=DB_COMPUTED Then
                        m2=m2 & " , " & "COMPUTED"
                End If
                If fieldinfo(DB_INFO_MONEY)=DB_MONEY Then
                        m2=m2 & " , " & "MONEY"
                End If
                If fieldinfo(DB_INFO_NULLABLE)=DB_NO_NULLS Then
                        m2=m2 & " , " & "DB_NO_NULLS"
                End If
                If fieldinfo(DB_INFO_NULLABLE)=DB_NULLABLE Then
                        m2=m2 & " , " & "DB_NULLABLE"
                End If
                If fieldinfo(DB_INFO_NULLABLE)=DB_NULLS_UNKNOWN Then
                        m2=m2 & " , " & "DB_NULLS_UNKNOWN"
                End If
                If fieldinfo(DB_INFO_READONLY)=DB_READONLY Then
                        m2=m2 & " , " & "READONLY"
                End If
                If fieldinfo(DB_INFO_READONLY)=DB_READWRITE Then
                        m2=m2 & " , " & "READWRITE"
                End If
                msg=msg & Chr(10) & i & " -> " & result.fieldname(i) & " , " & " size " & " " & result.fieldsize(i) & ", " &
m2
        Next
        Messagebox msg
        msg="Fields in " & dsn & " Table(Second Part of The Table):" & Chr(10)
        For i=1 To result.numcolumns
                fieldinfo=result.fieldinfo(i)
                m2=""
                m2=m2 & " , DB_COLUMNID= " & fieldinfo(DB_INFO_COLUMNID)
                m2=m2 & " , DB_COLUMNNAME= " & fieldinfo(DB_INFO_COLUMNNAME)
                m2=m2 & " , DB_DISPLAYSIZE= " & fieldinfo(DB_INFO_DISPLAYSIZE)
                m2=m2 & " , DB_EXPECTED_DATATYPE= " & fieldinfo(DB_INFO_EXPECTED_DATATYPE)
                m2=m2 & " , DB_LENGTH= " & fieldinfo(DB_INFO_LENGTH)
                msg=msg & Chr(10) & i & " -> " & result.fieldname(i) & " , " & " size " & " " & result.fieldsize(i) & ", " &
m2
        Next
        Messagebox msg
        msg="Fields in " & dsn & " Table(Third Part of The Table):" & Chr(10)
        For i=1 To result.numcolumns
                fieldinfo=result.fieldinfo(i)
                m2=""
                If fieldinfo(DB_INFO_READONLY)=DB_READONLY_UNKNOWN Then
                        m2=m2 & " , "  & "READONLY_UNKNOWN"
                End If
                If fieldinfo(DB_INFO_SEARCHABLE)=DB_SEARCHABLE Then
                        m2=m2 & " , " & "DB_SEARCHABLE"
                End If
                If fieldinfo(DB_INFO_SEARCHABLE)=DB_UNSEARCHABLE Then
                        m2=m2 & " , " & "DB_UNSEARCHABLE"
                End If
                If fieldinfo(DB_INFO_SEARCHABLE)=DB_LIKE_ONLY Then
                        m2=m2 & " , " & "DB_LIKE_ONLY"
                End If
                If fieldinfo(DB_INFO_SEARCHABLE)=DB_ALLEXCEPT_LIKE Then
                        m2=m2 & " , " & "DB_ALLEXCEPT_LIKE"
                End If
                If fieldinfo(DB_INFO_SETTABLE)=DB_SETTABLE Then
                        m2=m2 & " , " & "DB_SETTABLE"
                End If
                If fieldinfo(DB_INFO_UNSIGNED)=DB_UNSIGNED Then
                        m2=m2 & " , " & "DB_UNSIGNED"
                End If
                msg=msg & Chr(10) & i & " -> " & result.fieldname(i) & " , " & " size " & " " & result.fieldsize(i) & ", " &
m2
        Next
        Messagebox msg
        msg="Fields in " & dsn & " Table(Fourth Part of The Table):" & Chr(10)
        For i=1 To result.numcolumns
                fieldinfo=result.fieldinfo(i)
                m2=""
                m2=m2 & " , DB_NATIVE_DATATYPE= " & fieldinfo(DB_INFO_NATIVE_DATATYPE)
                m2=m2 & " , DB_PRECISION= " & fieldinfo(DB_INFO_PRECISION)
                m2=m2 & " , DB_SCALE= " & fieldinfo(DB_INFO_SCALE)
                m2=m2 & " , DB_SQLDATATYPE= " & fieldinfo(DB_INFO_SQLDATATYPE)
                m2=m2 & " , DB_TABLENAME= " & fieldinfo(DB_INFO_TABLENAME)
                If result.fieldnativedatatype(i)=SQL_CHAR Then
                        m2=m2 & " ,FieldNativeDataType= SQL_CHAR"
                End If
                If result.fieldnativedatatype(i)=SQL_NUMERIC Then
                        m2=m2 & " ,FieldNativeDataType= SQL_NUMERIC"
                End If
                If result.fieldnativedatatype(i)=SQL_DECIMAL Then
                        m2=m2 & " ,FieldNativeDataType= SQL_DECIMAL"
                End If
                If result.fieldnativedatatype(i)=SQL_INTEGER Then
                        m2=m2 & " ,FieldNativeDataType= SQL_INTEGER"
                End If
                If result.fieldnativedatatype(i)=SQL_SMALLINT Then
                        m2=m2 & " ,FieldNativeDataType= SQL_SMALLINT"
                End If
                If result.fieldnativedatatype(i)=SQL_FLOAT Then
                        m2=m2 & " ,FieldNativeDataType= SQL_FLOAT"
                End If
                If result.fieldnativedatatype(i)=SQL_REAL Then
                        m2=m2 & " ,FieldNativeDataType= SQL_REAL"
                End If
                If result.fieldnativedatatype(i)=SQL_REALSQL_DOUBLE Then
                        m2=m2 & " ,FieldNativeDataType= SQL_REALSQL_DOUBLE"
                End If
                If result.fieldnativedatatype(i)=SQL_DATE Then
                        m2=m2 & " ,FieldNativeDataType= SQL_DATE"
                End If
                If result.fieldnativedatatype(i)=SQL_TIME Then
                        m2=m2 & ",FieldNativeDataType= SQL_TIME"
                End If
                If result.fieldnativedatatype(i)=SQL_TIMESTAMP Then
                        m2=m2 & " ,FieldNativeDataType= SQL_TIMESTAMP"
                End If
                If result.fieldnativedatatype(i)=SQL_VARCHAR Then
                        m2=m2 & " ,FieldNativeDataType= SQL_VARCHAR"
                End If
                If result.fieldnativedatatype(i)=SQL_BINARY Then
                        m2=m2 & " ,FieldNativeDataType= SQL_BINARY"
                End If
                If result.fieldnativedatatype(i)=SQL_VARBINARY Then
                        m2=m2 & " ,FieldNativeDataType= SQL_VARBINARY"
                End If
                If result.fieldnativedatatype(i)=SQL_LONGVARCHAR Then
                        m2=m2 & " ,FieldNativeDataType= SQL_LONGVARCHAR"
                End If
                If result.fieldnativedatatype(i)=SQL_LONGVARBINARY Then
                        m2=m2 & " ,FieldNativeDataType= SQL_LONGVARBINARY"
                End If
```

```
                    If result.fieldnativedatatype(i)=SQL_BIGINT Then
                              m2=m2 & " ,FieldNativeDataType= SQL_BIGINT"
                    End If
                    If result.fieldnativedatatype(i)=SQL_TINYINT Then
                              m2=m2 & " ,FieldNativeDataType= SQL_TINYINT"
                    End If
                    If result.fieldnativedatatype(i)=SQL_BIT Then
                              m2=m2 & " ,FieldNativeDataType= SQL_BIT"
                    End If
                    msg=msg & Chr(10) & i & " -> " & result.fieldname(i) & " , " & " size " & " " & result.fieldsize(i) & ", " &
m2
          Next
          Messagebox msg
          result.close(DB_CLOSE)
          Call con.disconnect
End Sub
```

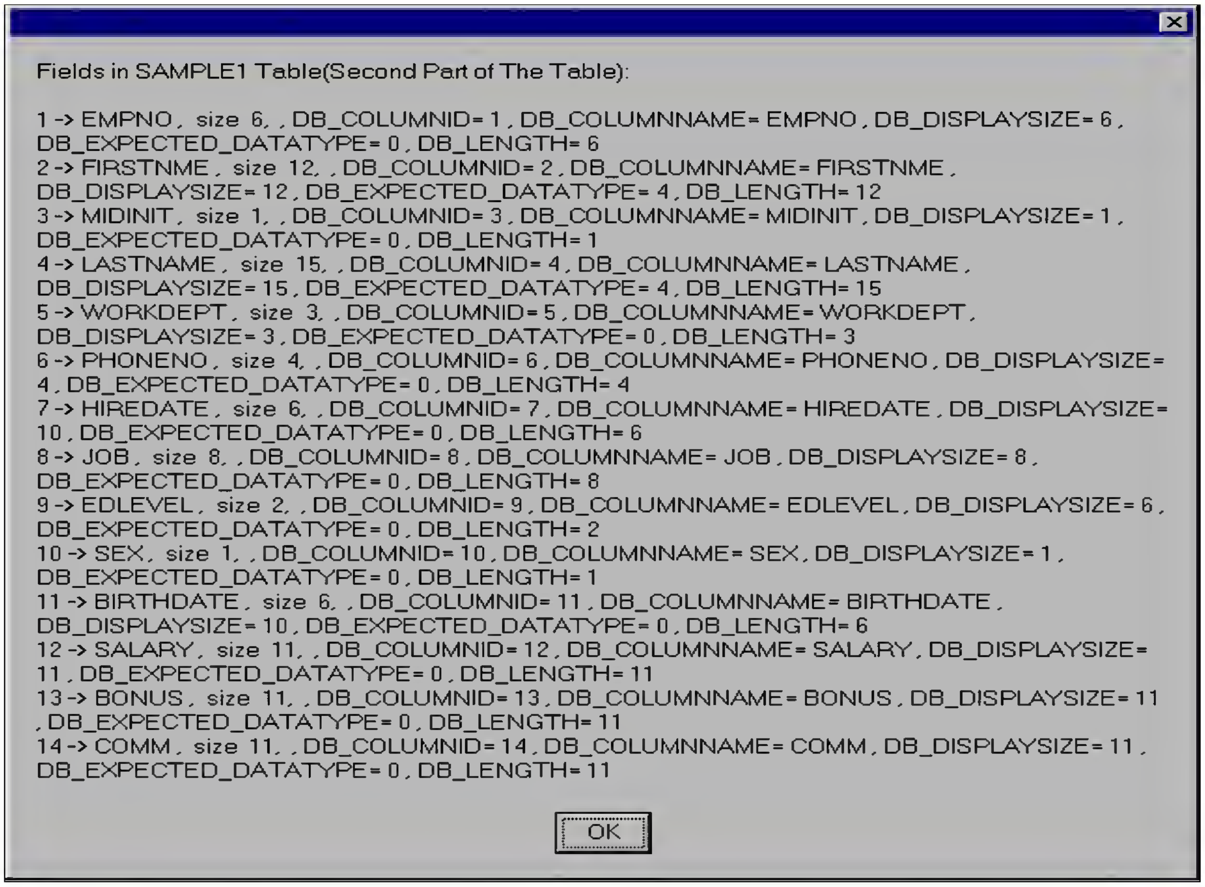
In order to run **AGENT5** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT5

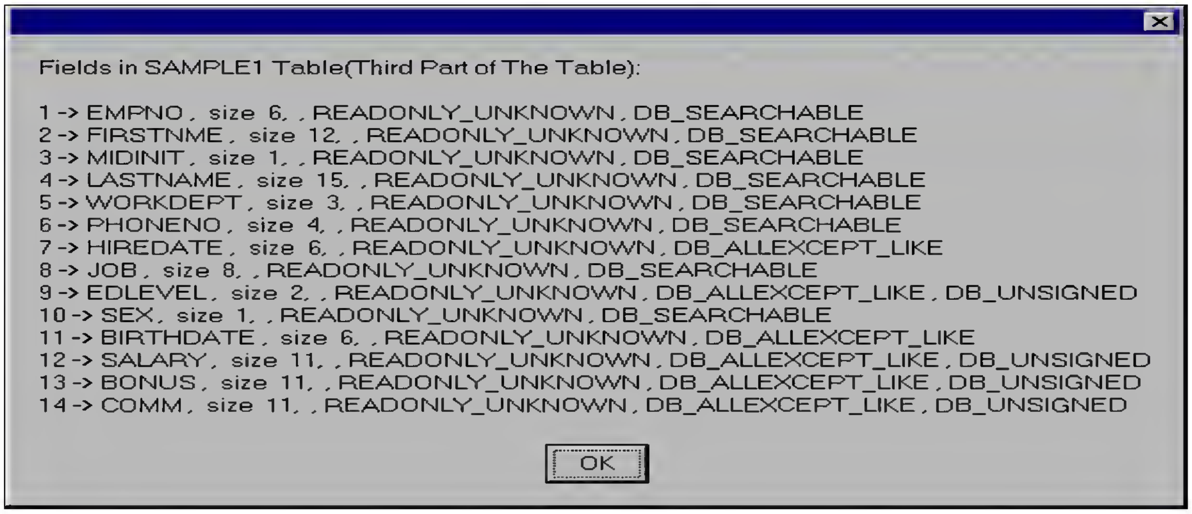
The result is as follows:



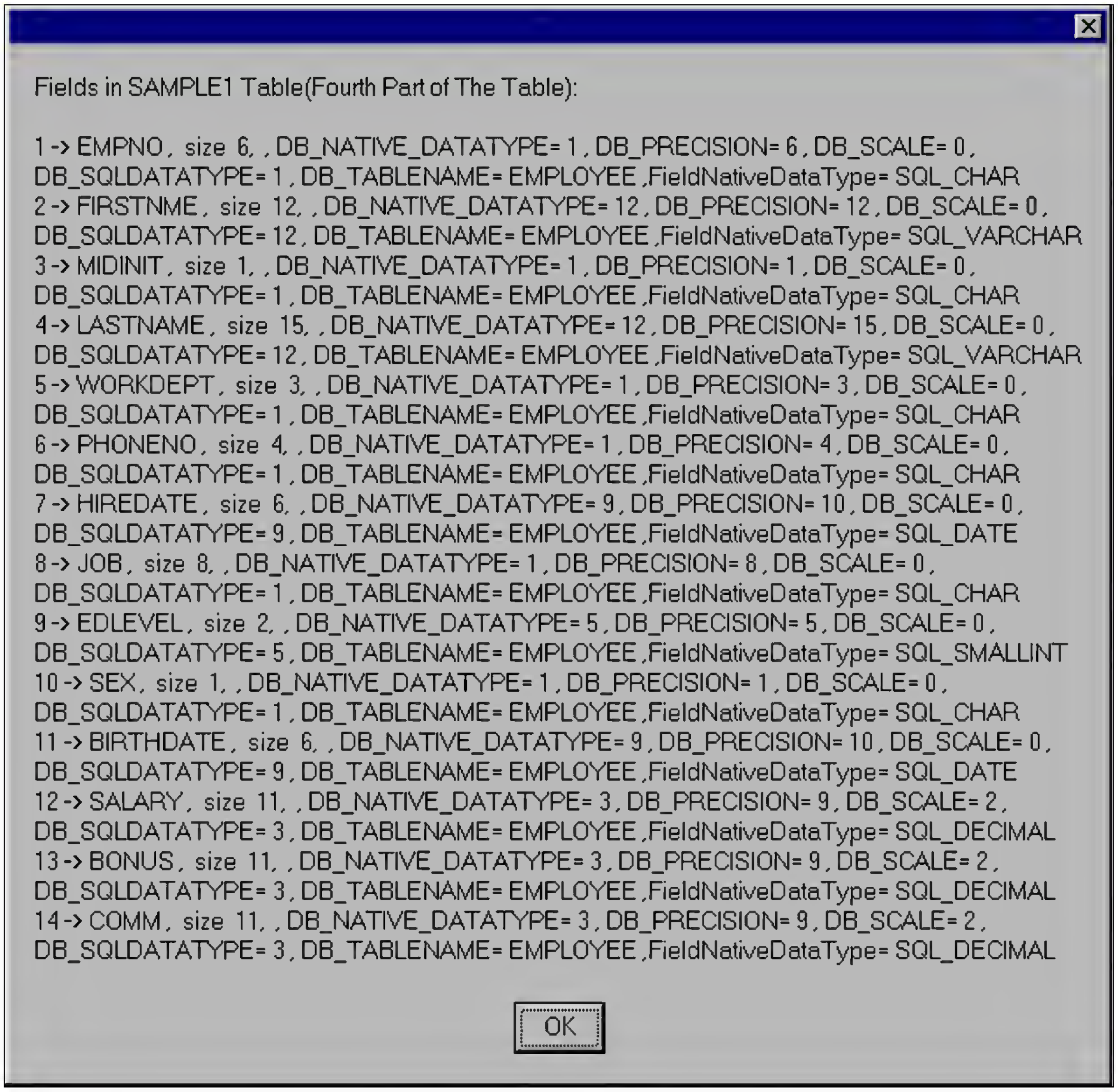
```
Fields in SAMPLE1 Table(First Part of The Table):

1 -> EMPNO , size 6, , AUTOINCREMENT , COMPUTED , MONEY , DB_NO_NULLS
2 -> FIRSTNME , size 12, , AUTOINCREMENT , COMPUTED , MONEY , DB_NO_NULLS
3 -> MIDINIT , size 1, , AUTOINCREMENT , COMPUTED , MONEY , DB_NO_NULLS
4 -> LASTNAME , size 15, , AUTOINCREMENT , COMPUTED , MONEY , DB_NO_NULLS
5 -> WORKDEPT , size 3, , AUTOINCREMENT , COMPUTED , MONEY , DB_NULLABLE
6 -> PHONENO , size 4, , AUTOINCREMENT , COMPUTED , MONEY , DB_NULLABLE
7 -> HIREDATE , size 6, , AUTOINCREMENT , CASESENSITIVE , COMPUTED , MONEY ,
DB_NULLABLE
8 -> JOB , size 8, , AUTOINCREMENT , COMPUTED , MONEY , DB_NULLABLE
9 -> EDLEVEL , size 2, , AUTOINCREMENT , CASESENSITIVE , COMPUTED , MONEY ,
DB_NO_NULLS
10 -> SEX , size 1, , AUTOINCREMENT , COMPUTED , MONEY , DB_NULLABLE
11 -> BIRTHDATE , size 6, , AUTOINCREMENT , CASESENSITIVE , COMPUTED , MONEY ,
DB_NULLABLE
12 -> SALARY , size 11, , AUTOINCREMENT , CASESENSITIVE , COMPUTED , MONEY ,
DB_NULLABLE
13 -> BONUS , size 11, , AUTOINCREMENT , CASESENSITIVE , COMPUTED , MONEY ,
DB_NULLABLE
14 -> COMM , size 11, , AUTOINCREMENT , CASESENSITIVE , COMPUTED , MONEY ,
DB_NULLABLE
```



```
Fields in SAMPLE1 Table(Second Part of The Table):

1 -> EMPNO , size 6, , DB_COLUMNID= 1 , DB_COLUMNNAME= EMPNO , DB_DISPLAYSIZE= 6 ,
DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 6
2 -> FIRSTNME , size 12, , DB_COLUMNID= 2 , DB_COLUMNNAME= FIRSTNME ,
DB_DISPLAYSIZE= 12 , DB_EXPECTED_DATATYPE= 4 , DB_LENGTH= 12
3 -> MIDINIT , size 1, , DB_COLUMNID= 3 , DB_COLUMNNAME= MIDINIT , DB_DISPLAYSIZE= 1 ,
DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 1
4 -> LASTNAME , size 15, , DB_COLUMNID= 4 , DB_COLUMNNAME= LASTNAME ,
DB_DISPLAYSIZE= 15 , DB_EXPECTED_DATATYPE= 4 , DB_LENGTH= 15
5 -> WORKDEPT , size 3, , DB_COLUMNID= 5 , DB_COLUMNNAME= WORKDEPT ,
DB_DISPLAYSIZE= 3 , DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 3
6 -> PHONENO , size 4, , DB_COLUMNID= 6 , DB_COLUMNNAME= PHONENO , DB_DISPLAYSIZE=
4 , DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 4
7 -> HIREDATE , size 6, , DB_COLUMNID= 7 , DB_COLUMNNAME= HIREDATE , DB_DISPLAYSIZE=
10 , DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 6
8 -> JOB , size 8, , DB_COLUMNID= 8 , DB_COLUMNNAME= JOB , DB_DISPLAYSIZE= 8 ,
DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 8
9 -> EDLEVEL , size 2, , DB_COLUMNID= 9 , DB_COLUMNNAME= EDLEVEL , DB_DISPLAYSIZE= 6 ,
DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 2
10 -> SEX, size 1, , DB_COLUMNID= 10 , DB_COLUMNNAME= SEX, DB_DISPLAYSIZE= 1 ,
DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 1
11 -> BIRTHDATE , size 6, , DB_COLUMNID= 11 , DB_COLUMNNAME= BIRTHDATE ,
DB_DISPLAYSIZE= 10 , DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 6
12 -> SALARY , size 11, , DB_COLUMNID= 12 , DB_COLUMNNAME= SALARY , DB_DISPLAYSIZE=
11 , DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 11
13 -> BONUS , size 11, , DB_COLUMNID= 13 , DB_COLUMNNAME= BONUS , DB_DISPLAYSIZE= 11
, DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 11
14 -> COMM , size 11, , DB_COLUMNID= 14 , DB_COLUMNNAME= COMM , DB_DISPLAYSIZE= 11 ,
DB_EXPECTED_DATATYPE= 0 , DB_LENGTH= 11
```



```
Fields in SAMPLE1 Table(Third Part of The Table):

1 -> EMPNO , size 6, , READONLY_UNKNOWN , DB_SEARCHABLE
2 -> FIRSTNME , size 12, , READONLY_UNKNOWN , DB_SEARCHABLE
3 -> MIDINIT , size 1, , READONLY_UNKNOWN , DB_SEARCHABLE
4 -> LASTNAME , size 15, , READONLY_UNKNOWN , DB_SEARCHABLE
5 -> WORKDEPT , size 3, , READONLY_UNKNOWN , DB_SEARCHABLE
6 -> PHONENO , size 4, , READONLY_UNKNOWN , DB_SEARCHABLE
7 -> HIREDATE , size 6, , READONLY_UNKNOWN , DB_ALLEXCEPT_LIKE
8 -> JOB , size 8, , READONLY_UNKNOWN , DB_SEARCHABLE
9 -> EDLEVEL , size 2, , READONLY_UNKNOWN , DB_ALLEXCEPT_LIKE , DB_UNSIGNED
10 -> SEX , size 1, , READONLY_UNKNOWN , DB_SEARCHABLE
11 -> BIRTHDATE , size 6, , READONLY_UNKNOWN , DB_ALLEXCEPT_LIKE
12 -> SALARY , size 11, , READONLY_UNKNOWN , DB_ALLEXCEPT_LIKE , DB_UNSIGNED
13 -> BONUS , size 11, , READONLY_UNKNOWN , DB_ALLEXCEPT_LIKE , DB_UNSIGNED
14 -> COMM , size 11, , READONLY_UNKNOWN , DB_ALLEXCEPT_LIKE , DB_UNSIGNED
```



```
Fields in SAMPLE1 Table(Fourth Part of The Table):

1 -> EMPNO , size 6, , DB_NATIVE_DATATYPE= 1 , DB_PRECISION= 6 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 1 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_CHAR
2 -> FIRSTNME , size 12, , DB_NATIVE_DATATYPE= 12 , DB_PRECISION= 12 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 12 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_VARCHAR
3 -> MIDINIT , size 1, , DB_NATIVE_DATATYPE= 1 , DB_PRECISION= 1 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 1 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_CHAR
4 -> LASTNAME , size 15, , DB_NATIVE_DATATYPE= 12 , DB_PRECISION= 15 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 12 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_VARCHAR
5 -> WORKDEPT , size 3, , DB_NATIVE_DATATYPE= 1 , DB_PRECISION= 3 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 1 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_CHAR
6 -> PHONENO , size 4, , DB_NATIVE_DATATYPE= 1 , DB_PRECISION= 4 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 1 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_CHAR
7 -> HIREDATE , size 6, , DB_NATIVE_DATATYPE= 9 , DB_PRECISION= 10 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 9 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_DATE
8 -> JOB , size 8, , DB_NATIVE_DATATYPE= 1 , DB_PRECISION= 8 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 1 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_CHAR
9 -> EDLEVEL , size 2, , DB_NATIVE_DATATYPE= 5 , DB_PRECISION= 5 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 5 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_SMALLINT
10 -> SEX , size 1, , DB_NATIVE_DATATYPE= 1 , DB_PRECISION= 1 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 1 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_CHAR
11 -> BIRTHDATE , size 6, , DB_NATIVE_DATATYPE= 9 , DB_PRECISION= 10 , DB_SCALE= 0 ,
DB_SQLDATATYPE= 9 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_DATE
12 -> SALARY , size 11, , DB_NATIVE_DATATYPE= 3 , DB_PRECISION= 9 , DB_SCALE= 2 ,
DB_SQLDATATYPE= 3 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_DECIMAL
13 -> BONUS , size 11, , DB_NATIVE_DATATYPE= 3 , DB_PRECISION= 9 , DB_SCALE= 2 ,
DB_SQLDATATYPE= 3 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_DECIMAL
14 -> COMM , size 11, , DB_NATIVE_DATATYPE= 3 , DB_PRECISION= 9 , DB_SCALE= 2 ,
DB_SQLDATATYPE= 3 , DB_TABLENAME= EMPLOYEE ,FieldNativeDataType= SQL_DECIMAL
```

Example 3.6

This example shows an agent (AGENT6) that accesses all the rows of a result set twice, starting from the first row. The first time you do not explicitly set FirstRow since the first NextRow following an EXECUTE implicitly sets FirstRow. The second time, you must explicitly set FirstRow and process the first row before entering the loop.
In order to achieve this objective do the following steps:

Step A - 3.6

Create the agent AGENT6 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.6

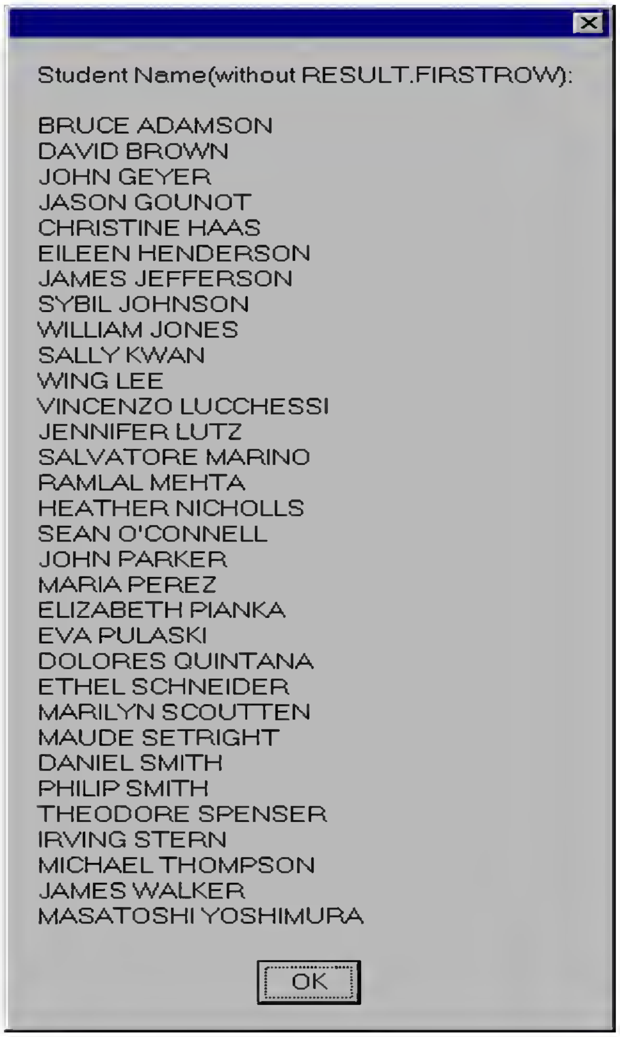
Create the following LotusScript code for AGENT6:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim firstname As String
        Dim lastname As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                Messagebox "Could not connect to " & dsn & " DataBase"
                End
        End If
        qry.sql="select * from EMPLOYEE order by LASTNAME"
        result.execute
        If Not result.isresultsetavailable Then
                Messagebox "Couldn't get result set"
                Call con.disconnect
                End
        End If
        msg="Student Name(without RESULT.FIRSTROW): " & Chr(10)
        Do
                result.nextrow
                firstname=result.getvalue("FIRSTNME",firstname)
                lastname=result.getvalue("LASTNAME",lastname)
                msg=msg & Chr(10) & firstname & " " & lastname
        Loop Until result.isendofdata
        Messagebox msg
        msg="Student Name(with RESULT.FIRSTROW): " & Chr(10)
        result.firstrow
        firstname=result.getvalue("FIRSTNME",firstname)
        lastname=result.getvalue("LASTNAME",lastname)
        msg=msg & Chr(10) & firstname & " " & lastname
        Do
                result.nextrow
                firstname=result.getvalue("FIRSTNME",firstname)
                lastname=result.getvalue("LASTNAME",lastname)
                msg=msg & Chr(10) & firstname & " " & lastname
        Loop Until result.isendofdata
        Messagebox msg
        result.close(DB_CLOSE)
        con.disconnect
End Sub
```

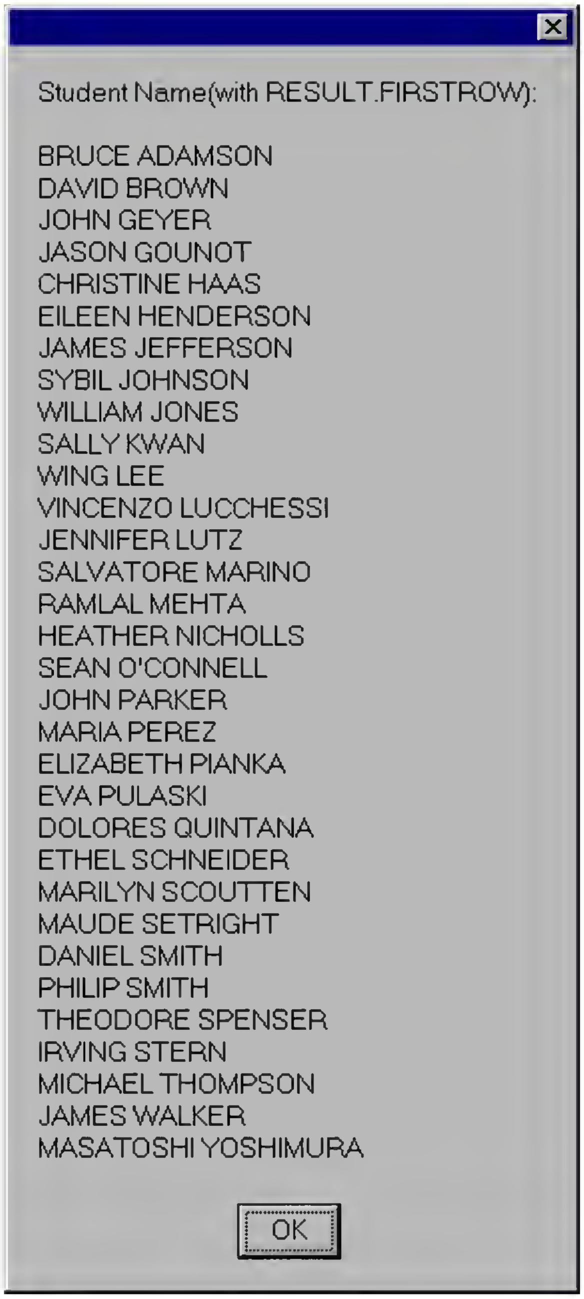
In order to run **AGENT6** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT6

The result is as follows:



```
Student Name(without RESULT.FIRSTROW):

BRUCE ADAMSON
DAVID BROWN
JOHN GEYER
JASON GOUNOT
CHRISTINE HAAS
EILEEN HENDERSON
JAMES JEFFERSON
SYBIL JOHNSON
WILLIAM JONES
SALLY KWAN
WING LEE
VINCENZO LUCCHESSI
JENNIFER LUTZ
SALVATORE MARINO
RAMLAL MEHTA
HEATHER NICHOLLS
SEAN O'CONNELL
JOHN PARKER
MARIA PEREZ
ELIZABETH PIANKA
EVA PULASKI
DOLORES QUINTANA
ETHEL SCHNEIDER
MARILYN SCOUTTEN
MAUDE SETRIGHT
DANIEL SMITH
PHILIP SMITH
THEODORE SPENSER
IRVING STERN
MICHAEL THOMPSON
JAMES WALKER
MASATOSHI YOSHIMURA
```



```
Student Name(with RESULT.FIRSTROW):

BRUCE ADAMSON
DAVID BROWN
JOHN GEYER
JASON GOUNOT
CHRISTINE HAAS
EILEEN HENDERSON
JAMES JEFFERSON
SYBIL JOHNSON
WILLIAM JONES
SALLY KWAN
WING LEE
VINCENZO LUCCHESSI
JENNIFER LUTZ
SALVATORE MARINO
RAMLAL MEHTA
HEATHER NICHOLLS
SEAN O'CONNELL
JOHN PARKER
MARIA PEREZ
ELIZABETH PIANKA
EVA PULASKI
DOLORES QUINTANA
ETHEL SCHNEIDER
MARILYN SCOUTTEN
MAUDE SETRIGHT
DANIEL SMITH
PHILIP SMITH
THEODORE SPENSER
IRVING STERN
MICHAEL THOMPSON
JAMES WALKER
MASATOSHI YOSHIMURA
```

Example 3.7

This example locates all the rows in a result set with **"JAMES"** in **"FIRSTNME"** field and **"DESIGNER** in field **2**.
In order to achieve this objective do the following steps:

Step A - 3.7

Create the agent AGENT7 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in Database.

Step B - 3.7

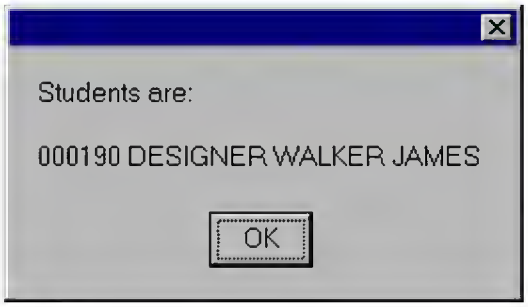
Create the following LotusScript code for AGENT:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                Messagebox "Could not connect to " & dsn & " DataBase"
                End
        End If
        qry.sql="select EMPNO, JOB, LASTNAME, FIRSTNME from EMPLOYEE order by LASTNAME"
        result.execute
        If Not result.isresultsetavailable Then
                Messagebox "Couldn't get result set"
                Call con.disconnect
                End
        End If
        msg="Students are:" & Chr(10)
        result.firstrow
        Do While result.locaterow("FIRSTNME", "JAMES",2, "DESIGNER")
                msg=msg & Chr(10)
                For i=1 To result.numcolumns
                        msg=msg & result.getvalue(i) & " "
                next
                If result.isendofdata Then Exit Do
                result.nextrow
        Loop
        Messagebox msg
        result.close(DB_CLOSE)
        con.disconnect
End Sub
```

In order to run **AGENT7** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT7

The result is as follows:



```
Students are:

000190 DESIGNER WALKER JAMES
```

Example 3.8

This example displays all rows in EMPLOYEE table, for each row showing the values of EMPNO, FIRSTNME, LASTNAME. The variable into which the result set value is stored, is also used as the second argument to GetValue in order to make the data typing explicitly.
In order to achieve this objective do the following steps:

<u>Step A - 3.8</u>

Create the agent AGENT8 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

<u>Step B - 3.8</u>

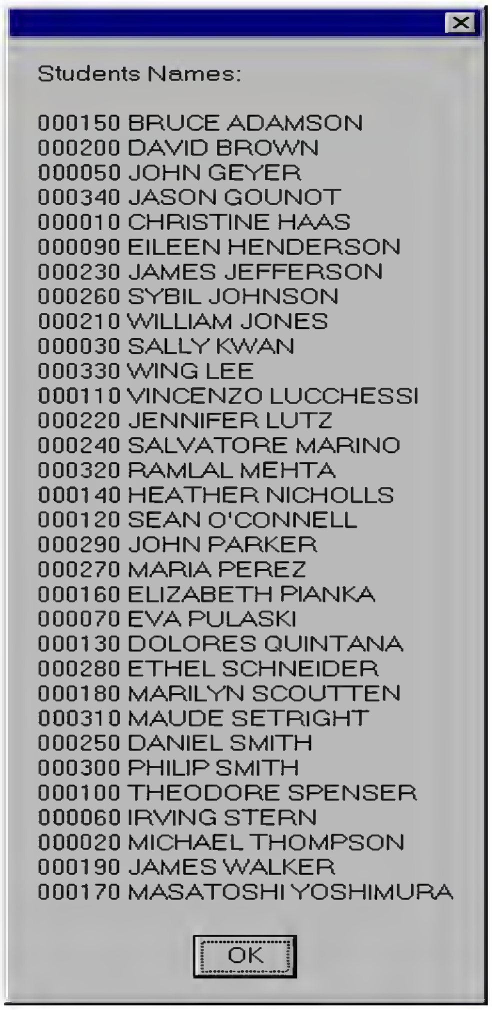
Create the following LotusScript code for AGENT8:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim studentno As String
        Dim firstname As String
        Dim lastname As String
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                Messagebox "Could not connect to " & dsn & " DataBase"
                End
        End If
        qry.sql="select * from EMPLOYEE order by LASTNAME"
        result.execute
        If Not result.isresultsetavailable Then
                Messagebox "Couldn't get result set"
                Call con.disconnect
                End
        End If
        msg="Students Names:" & Chr(10)
        Do
                result.nextrow
                If result.isvaluenull("EMPNO") Then
                        studentno="None"
                Else
                        studentno=result.getvalue("EMPNO",studentno)
                End If
                If result.isvaluenull("FIRSTNME") Then
                        firstname="None"
                Else
                        firstname=result.getvalue("FIRSTNME",firstname)
                End If
                If result.isvaluenull("FIRSTNME") Then
                        lastname="None"
                Else
                        lastname=result.getvalue("LASTNAME",lastname)
                End If
                msg=msg & Chr(10) & studentno & " " & firstname & " " & lastname
        Loop Until result.isendofdata
        Messagebox msg
        Call con.disconnect
End Sub
```

In order to run **AGENT8** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT8

The result is as follows:



```
Students Names:

000150 BRUCE ADAMSON
000200 DAVID BROWN
000050 JOHN GEYER
000340 JASON GOUNOT
000010 CHRISTINE HAAS
000090 EILEEN HENDERSON
000230 JAMES JEFFERSON
000260 SYBIL JOHNSON
000210 WILLIAM JONES
000030 SALLY KWAN
000330 WING LEE
000110 VINCENZO LUCCHESSI
000220 JENNIFER LUTZ
000240 SALVATORE MARINO
000320 RAMLAL MEHTA
000140 HEATHER NICHOLLS
000120 SEAN O'CONNELL
000290 JOHN PARKER
000270 MARIA PEREZ
000160 ELIZABETH PIANKA
000070 EVA PULASKI
000130 DOLORES QUINTANA
000280 ETHEL SCHNEIDER
000180 MARILYN SCOUTTEN
000310 MAUDE SETRIGHT
000250 DANIEL SMITH
000300 PHILIP SMITH
000100 THEODORE SPENSER
000060 IRVING STERN
000020 MICHAEL THOMPSON
000190 JAMES WALKER
000170 MASATOSHI YOSHIMURA
```

Example 3.9

This example displays, just for the first row of EMPLOYEE table, the name of column, the type of column and the value of column.
In order to achieve this objective do the following steps:

Step A - 3.9

Create the agent AGENT9 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.9

Create the following LotusScript code for AGENT9:

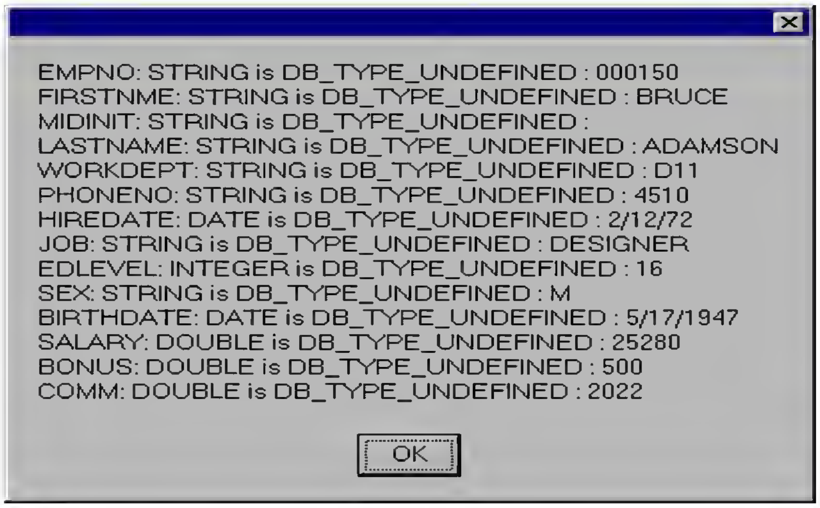
```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                Messagebox "Could not connect to " & dsn & " DataBase"
                End
        End If
        qry.sql="select * from EMPLOYEE order by LASTNAME"
        result.execute
        If Not result.isresultsetavailable Then
                Messagebox "Couldn't get result set"
                Call con.disconnect
                End
        End If
        result.nextrow
        msg=""
        For i=1 To result.numcolumns
                If (result.fieldexpecteddatatype(i) = DB_TYPE_UNDEFINED) Then
                        msg=msg & result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is
DB_TYPE_UNDEFINED" & " : " & result.getvalue(i) & Chr(10)
                End If
                If (result.fieldexpecteddatatype(i) = DB_CHAR) Then
                        msg=msg & result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_CHAR" & " : " &
result.getvalue(i) & Chr(10)
                End If
                If (result.fieldexpecteddatatype(i) = DB_SHORT) Then
                        msg=msg &  result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_SHORT" & " : "
& result.getvalue(i) & Chr(10)
                End If
```

```
                    If (result.fieldexpecteddatatype(i) = DB_LONG) Then
                              msg=msg &   result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_LONG" & " : "
& result.getvalue(i) & Chr(I0)
                    End If
                    If (result.fieldexpecteddatatype(i) = DB_DOUBLE) Then
                              msg=msg &   result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_DOUBLE" & " :
" & result.getvalue(i) & Chr(10)
                    End If
                    If (result.fieldexpecteddatatype(i) = DB_DATE) Then
                              msg=msg & result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_DATE" & " : " &
result.getvalue(i) & Chr(10)
                    End If
                    If (result.fieldexpecteddatatype(i) = DB_TIME) Then
                              msg=msg &  result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_TIME" & " : " &
result.getvalue(i) & Chr(10)
                    End If
                    If (result.fieldexpecteddatatype(i) = DB_BINARY) Then
                              msg=msg &  result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_BINARY" & " :
" & result.getvalue(i) & Chr(10)
                    End If
                    If (result.fieldexpecteddatatype(i) = DB_BOOL) Then
                              msg=msg &  result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_BOOL" & " : "
& result.getvalue(i) & Chr(I0)
                    End If
                    If (result.fieldexpecteddatatype(i) = DB_DATETIME) Then
                              msg=msg &  result.fieldname(i) & ": " & Typename(result.getvalue(i)) & " is DB_DATETIME" &
" : " & result.getvalue(i) & Chr(I0)
                    End If
          Next
          Messagebox msg
          result.close(DB_CLOSE)
          con.disconnect
End Sub
```

In order to run **AGENT9** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT9

The result is as follows:



```
EMPNO: STRING is DB_TYPE_UNDEFINED : 000150
FIRSTNME: STRING is DB_TYPE_UNDEFINED : BRUCE
MIDINIT: STRING is DB_TYPE_UNDEFINED :
LASTNAME: STRING is DB_TYPE_UNDEFINED : ADAMSON
WORKDEPT: STRING is DB_TYPE_UNDEFINED : D11
PHONENO: STRING is DB_TYPE_UNDEFINED : 4510
HIREDATE: DATE is DB_TYPE_UNDEFINED : 2/12/72
JOB: STRING is DB_TYPE_UNDEFINED : DESIGNER
EDLEVEL: INTEGER is DB_TYPE_UNDEFINED : 16
SEX: STRING is DB_TYPE_UNDEFINED : M
BIRTHDATE: DATE is DB_TYPE_UNDEFINED : 5/17/1947
SALARY: DOUBLE is DB_TYPE_UNDEFINED : 25280
BONUS: DOUBLE is DB_TYPE_UNDEFINED : 500
COMM: DOUBLE is DB_TYPE_UNDEFINED : 2022
```

Example 3.10

This example is based on a form and view, both named "PhoneBook. The form has three fields: lastName, firstName, phoneNumber. The view has seven Actions. The example also uses the agent AGENT11.
The following items are exercised:

- ACTION1: creates new table onto DB2 (named Phone), deletes a table (named Phone) adds new rows into the Phone table.
- ACTION2: adds new rows into the Phone table.
- ACTION3: deletes a row into the Phone table but if the row is unique only; that means there aren't two columns in the Phone table having the same LASTNAME, FIRSTNAME.
- ACTION4: displays all rows of the Phone table using the sequence:

```
DO
    RESULT.NEXTROW
.
.
LOOP UNTIL RESULT.ISENDOFDATA
```

- ACTION5: DROPs the table Phone
- ACTION6: updates the column FIRSTNAME for the row FLOREA COSTICA 123456, changing COSTICA with CRISTINA
- ACTION7: displays all the rows of the Phone table using the sequence:

```
RESULT.LASTROW
FOR I=1 to RESULT.NUMROWS
    .
    .
NEXT
```

- AGENT11: deletes all rows from the Phone table, emptying the Phone table, but does not remove the Phone table. ACTION5 removes the Phone table.

In order to achieve this objective do the following steps:

Step A - 3.10

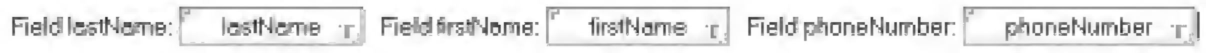
Create the agent AGENT11 having the features: Shared Agent, Manually From Action Menu, Should Act on All Documents in DataBase.

Step B - 3.10

Create the following LotusScript code for AGENT11:

```
Sub Initialize
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="delete from phone"
        If qry.geterror <> DBstsSUCCESS Then
                msg=msg & "ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & con.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
                End
        End If
        If Not result.execute() Then
                msg="Couldn't delete" & Chr(10)
                If result.geterror <> DBstsSUCCESS Then
                        msg=msg & "Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
                        msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                End If
                Messagebox msg
                con.disconnect
                End
        End If
        result.close(DB_CLOSE)
        con.disconnect
        Messagebox "Finish DELETE"
End Sub
```

Step C - 3.10

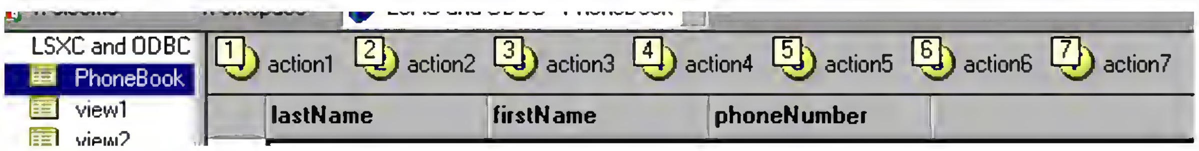


Create the form PhoneBook having the following fields (text + editable):
LastName, firstName, phoneName.

Step D - 3.10

Create the view PhoneBook each column of it being the image of fields from PhoneBook form, and having the following features:

```
Globals->Options:
                        Option Public
                        USELSX "*LSXODBC"
Globals->Declarations:
                        %INCLUDE "lsconst.lss"
                        Dim session As notessession
                        Dim db As notesdatabase
                        Dim view As notesview

Sub Postopen(Source As Notesuiview)
        Set session=New notessession
        Set db=session.currentdatabase
        Set view=db.getview("PhoneBook")
End sub
```



Step E - 3.10

Create the following LotusScript code for ACTION1:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="create table Phone (LASTNAME char(32), FIRSTNAME char(32), PHONENO char(16))"
        If qry.geterror <> DBstsSUCCESS Then
                msg="Error first qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
```

```
            End
        End If
        result.execute
        If result.geterror <> DBstsSUCCESS Then
            msg="Error first result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
            msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
            Messagebox msg
            If Messagebox ("Do you want to delete the existing table ?", MB_YESNO, "Table already exists")=IDYES
Then
                result.close(DB_CLOSE)
                qry.sql="DROP TABLE Phone"
                If Not result.execute() Then
                    msg="Couldn't drop" & Chr(10)
                    msg=msg & "Error first result.execute: ExtendedErrorMessage= " &
result.getextendederrormessage
                    msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                    Messagebox msg
                    con.disconnect
                    End
                End If
                result.close(DB_CLOSE)
                qry.sql="create table Phone (LASTNAME char(32), FIRSTNAME char(32), PHONENO
char(16))"
                If qry.geterror <> DBstsSUCCESS Then
                    msg="Error second qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
                    msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
                    Messagebox msg
                    End
                End If
                result.execute
                If result.geterror <> DBstsSUCCESS Then
                    msg="Error second result.execute: ExtendedErrorMessage= " &
result.getextendederrormessage
                    msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                    Messagebox msg
                    End
                End If
            Else
                result.close(DB_CLOSE)
                con.disconnect
                End
            End If
        End If
        If qry.geterror <> DBstsSUCCESS Then
            msg="Cannot run qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
            msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
            Messagebox msg
            result.close(DB_CLOSE)
            con.disconnect
            End
        End If
        result.close(DB_CLOSE)
        qry.SQL="select * from Phone"
        result.execute
        Set doc=view.getfirstdocument
        While Not (doc Is Nothing)
            result.addrow
            Call result.setvalue("LASTNAME", doc.lastName(0))
            Call result.setvalue("FIRSTNAME",doc.firstName(0))
            Call result.setvalue("PHONENO",doc.phoneNumber(0))
            result.updaterow
```

```
                Set doc=view.getnextdocument(doc)
        Wend
        result.close(DB_CLOSE)
        con.disconnect
        Messagebox "Finish ACTION1"
End Sub
```

Step F- 3.10

Create the following LotusScript code for ACTION2:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="select * from Phone"
        If qry.geterror <> DBstsSUCCESS Then
                msg="Error qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
                End
        End If
        result.execute
        If result.geterror <> DBstsSUCCESS Then
                msg="Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
        End If
        Set dc=db.unprocesseddocuments
        If dc.count=0 Then
                result.close(DB_CLOSE)
                con.disconnect
                Messagebox "There aren't UnprocessedDocuments"
                End
        End If
        For i=1 To dc.count
                Set doc=dc.getnthdocument(i)
                Call session.updateprocesseddoc(doc)
                result.addrow
                Call result.setvalue("LASTNAME", doc.lastName(0))
                Call result.setvalue("FIRSTNAME",doc.firstName(0))
                Call result.setvalue("PHONENO",doc.phoneNumber(0))
                result.updaterow
        Next
        result.close(DB_CLOSE)
        con.disconnect
        Messagebox "Finish ACTION2"
End Sub
```

Step G - 3.10

Create the following LotusScript code for ACTION3:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="select * from Phone"
        If qry.geterror <> DBstsSUCCESS Then
                msg="Error qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
                End
        End If
        result.execute
        If result.geterror <> DBstsSUCCESS Then
                msg="Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
        End If
        Set dc=db.unprocesseddocuments
        If dc.count=0 Then
                result.close(DB_CLOSE)
                con.disconnect
                Messagebox "There aren't UnprocessedDocuments"
                End
        End If
        For i=1 To dc.count
                Set doc=dc.getnthdocument(i)
                Call session.updateprocesseddoc(doc)
                If result.locaterow(1, doc.lastName(0), 2, doc.firstName(0)) Then
                        If result.geterror <> DBstsSUCCESS Then
                                msg="Error result.locaterow: ExtendedErrorMessage= " &
result.getextendederrormessage
                                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                                Messagebox msg
                                End
                        End If
                        result.deleterow("Phone")
                        If result.geterror <> DBstsSUCCESS Then
                                msg="Error result.deleterow: ExtendedErrorMessage= " &
result.getextendederrormessage
                                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                                Messagebox msg
                                End
                        End If
                End If
        Next
        view.refresh
        result.close(DB_CLOSE)
        con.disconnect
        Messagebox "Finish ACTION3"
End Sub
```

Step H - 3.10

Create the following LotusScript code for ACTION4:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim firstname As String
        Dim lastname As String
        Dim phoneno As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="select * from Phone order by LASTNAME"
        If qry.geterror <> DBstsSUCCESS Then
                msg="Error qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
                End
        End If
        result.execute
        If result.geterror <> DBstsSUCCESS Then
                msg="Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
        End If
        msg=""
        Call displayresultsetproperties(result,msg)
        msg=msg & Chr(10) & "Phone entries:"
        Do
                result.nextrow
                firstname=result.getvalue("FIRSTNAME",firstname)
                lastname=result.getvalue("LASTNAME",lastname)
                phoneno=result.getvalue("PHONENO",phoneno)
                msg=msg & Chr(10) & firstname & " " & lastname & "  " & phoneno
        Loop Until result.isendofdata
        msg=msg & Chr(10)
        Call displayresultsetproperties(result,msg)
        Messagebox msg
        result.close(DB_CLOSE)
        con.disconnect

End Sub

Sub displayresultsetproperties(result,msg)
        If result.isresultsetavailable Then
                If result.numrows=DB_NORESULT Then
                        msg=msg & Chr(10) & " result.numrows= DB_NORESULT"
                End If
                If result.numrows=DB_ROWSUNKNOWN Then
                        msg=msg & Chr(10) & " result.numrows= DB_ROWSUNKNOWN"
                End If
                If result.numrows=DB_ROWSLIMITED Then
                        msg=msg & Chr(10) & " result.numrows= DB_ROWSLIMITED"
                End If
                rows$=Cstr(result.numrows)
                msg=msg & Chr(10) & "NumColumns= " & result.numcolumns & Chr(10)  _
                & "NumRows= " & rows$ & Chr(10)  _
                & "IsBeginOfData= " & result.isbeginofdata & Chr(10) _
                & "IsEndOfData= " & result.isendofdata & Chr(10) _
                & "CurrentRow= " & result.currentrow & Chr(10)
        Else
                msg=msg & " Result set not available" & Chr(10)
        End If
End Sub
```

Step I - 3.10

Create the following LotusScript code for ACTION5:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="DROP TABLE Phone"
        If qry.geterror <> DBstsSUCCESS Then
                msg=msg & "ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & con.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
                End
        End If
        If Not result.execute() Then
                msg="Couldn't drop" & Chr(10)
                msg=msg & "Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                con.disconnect
                End
        End If
        result.close(DB_CLOSE)
        con.disconnect
        Messagebox "Finish ACTION5"
End Sub
```

Step J - 3.10

Create the following LotusScript code for ACTION6:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
```

```
    Call con.disconnect
    Call con.connectto(dsn,userid,parola)
    If Not con.isconnected Then
        msg="Could not connect to " & dsn & " DataBase" & Chr(10)
        If con.geterror <> DBstsSUCCESS Then
            msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
            msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
        End If
        Messagebox msg
        End
    End If
    qry.sql="select * from Phone"
    If qry.geterror <> DBstsSUCCESS Then
        msg="Error qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
        msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
        Messagebox msg
        End
    End If
    result.execute
    If result.geterror <> DBstsSUCCESS Then
        msg="Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
        msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
        Messagebox msg
        End
    End If
    Set dc=db.unprocesseddocuments
    If dc.count=0 Then
        result.close(DB_CLOSE)
        con.disconnect
        Messagebox "There aren't UnprocessedDocuments"
        End
    End If
    For i=1 To dc.count
        Set doc=dc.getnthdocument(i)
        Call session.updateprocesseddoc(doc)
        If result.locaterow(1, doc.lastName(0)) Then
            If result.geterror <> DBstsSUCCESS Then
                msg="Error result.locaterow: ExtendedErrorMessage= " &
result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
            End If
            Call result.setvalue(2, doc.firstName(0))
            If result.geterror <> DBstsSUCCESS Then
                msg="Error result.setvalue: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
            End If
            Call result.updaterow
            If result.geterror <> DBstsSUCCESS Then
                msg="Error result.update: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
            End If
        End If
    Next
    view.refresh
    result.close(DB_CLOSE)
```

```
        con.disconnect
        Messagebox "Finish ACTION6"
End Sub
```

Step K - 3.10

Create the following LotusScript code for ACTION7:

```
Sub Click(Source As Button)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim result As New odbcresultset
        Dim msg As String
        Dim firstname As String
        Dim lastname As String
        Dim phoneno As String
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        qry.sql="select * from Phone order by LASTNAME"
        If qry.geterror <> DBstsSUCCESS Then
                msg="Error qry.SQL: ExtendedErrorMessage= " & qry.getextendederrormessage
                msg=msg & " Error= " & qry.geterror & " ErrorMessage= " & qry.geterrormessage
                Messagebox msg
                End
        End If
        result.execute
        If result.geterror <> DBstsSUCCESS Then
                msg="Error result.execute: ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
                End
        End If
        msg=""
        Call displayresultsetproperties(result,msg)
        msg=msg & Chr(10) & "Phone entries:"
        result.lastrow
        For i=1 To result.numrows
                result.currentrow=i
                firstname=result.getvalue("FIRSTNAME",firstname)
                lastname=result.getvalue("LASTNAME",lastname)
                phoneno=result.getvalue("PHONENO",phoneno)
                msg=msg & Chr(10) & firstname & " " & lastname & "  " & phoneno
```

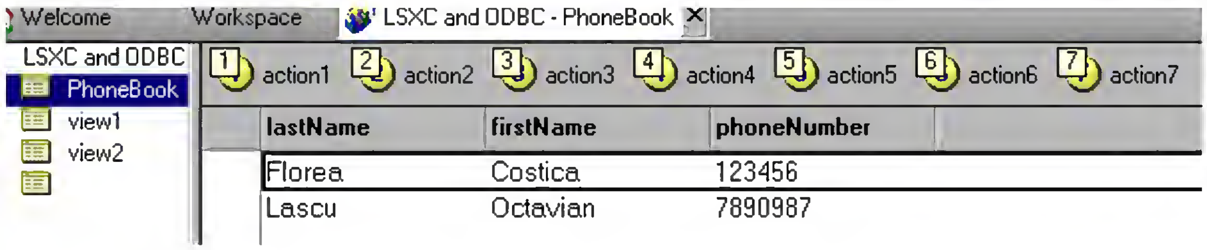
```
		Next
		msg=msg & Chr(10)
		Call displayresultsetproperties(result,msg)
		Messagebox msg
		result.close(DB_CLOSE)
		con.disconnect
End Sub

Sub displayresultsetproperties(result,msg)
		If result.isresultsetavailable Then
			If result.numrows=DB_NORESULT Then
				msg=msg & Chr(10) & " result.numrows= DB_NORESULT"
			End If
			If result.numrows=DB_ROWSUNKNOWN Then
				msg=msg & Chr(10) & " result.numrows= DB_ROWSUNKNOWN"
			End If
			If result.numrows=DB_ROWSLIMITED Then
				msg=msg & Chr(10) & " result.numrows= DB_ROWSLIMITED"
			End If
			rows$=Cstr(result.numrows)
			msg=msg & Chr(10) & "NumColumns= " & result.numcolumns & Chr(10) _
			& "NumRows= " & rows$ & Chr(10) _
			& "IsBeginOfData= " & result.isbeginofdata & Chr(10) _
			& "IsEndOfData= " & result.isendofdata & Chr(10) _
			& "CurrentRow= " & result.currentrow & Chr(10)
		Else
			msg=msg & " Result set not available" & Chr(10)
		End If
End Sub
```

In order to run **ACTION1** do the following steps:

✓ Select LSXCODBC.NSF DataBase ---> Create ---> PhoneBook and create two documents.
✓ Open the view PhoneBook and push onto ACTION1

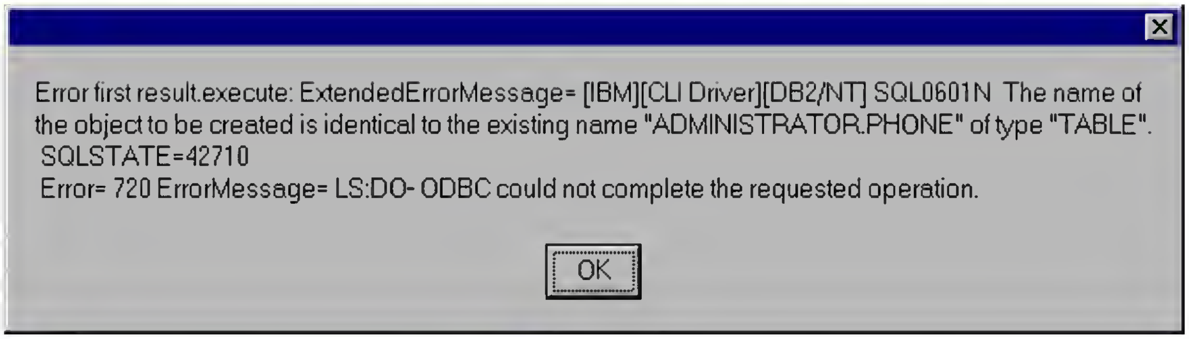
The result is as follows:

For view PhoneBook:



lastName	firstName	phoneNumber
Florea	Costica	123456
Lascu	Octavian	7890987

For ACTION1:



Error first result.execute: ExtendedErrorMessage= [IBM][CLI Driver][DB2/NT] SQL0601N The name of the object to be created is identical to the existing name "ADMINISTRATOR.PHONE" of type "TABLE". SQLSTATE=42710
Error= 720 ErrorMessage= LS:DO- ODBC could not complete the requested operation.

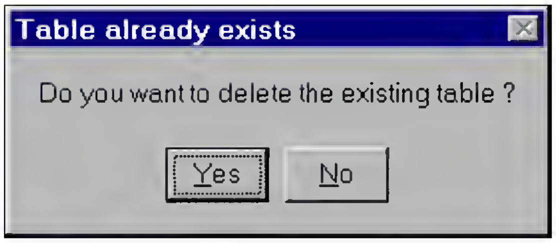
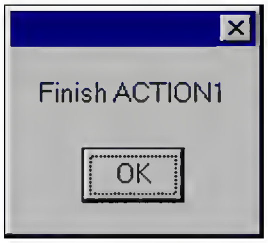


Table already exists

Do you want to delete the existing table ?

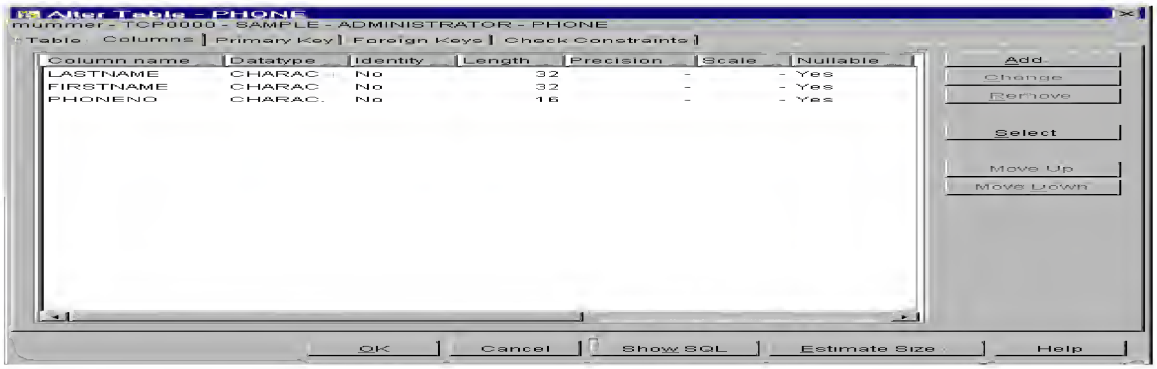


Finish ACTION1

The Content of Phone table in SAMPLE DataBase:

```
LASTNAME          FIRSTNAME         PHONENO
-----------------------------------------------------
Florea            Costica           123456
Lascu             Octavian          7890987
```

The Structure of the table Phone:



Alter Table - PHONE

mummer - TCP0000 - SAMPLE - ADMINISTRATOR - PHONE

Column name	Datatype	Identity	Length	Precision	Scale	Nullable
LASTNAME	CHARAC	No	32	-	-	Yes
FIRSTNAME	CHARAC	No	32	-	-	Yes
PHONENO	CHARAC.	No	16	-	-	Yes

In order to run **ACTION2** do the following steps:

- ✓ Select LSXCODBC.NSF DataBase ---> Create ---> PhoneBook and create some documents.
- ✓ Open the view PhoneBook, select some documents and push onto ACTION2.

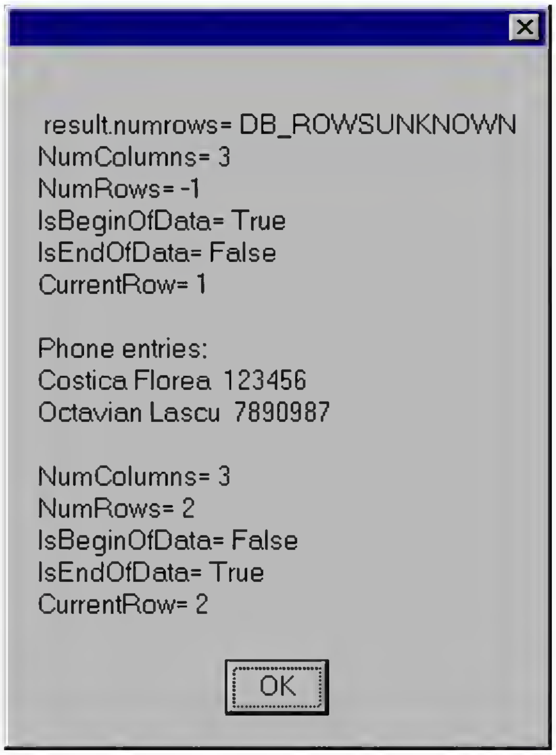
In order to run **ACTION3** do the following step:

- ✓ Open the view PhoneBook, select some documents and push onto **ACTION3.**

In order to run **ACTION4** do the following step:

- ✓ Open the view PhoneBook, select some documents and push onto **ACTION4.**

The result is as follows:



```
result.numrows= DB_ROWSUNKNOWN
NumColumns= 3
NumRows= -1
IsBeginOfData= True
IsEndOfData= False
CurrentRow= 1

Phone entries:
Costica Florea  123456
Octavian Lascu  7890987

NumColumns= 3
NumRows= 2
IsBeginOfData= False
IsEndOfData= True
CurrentRow= 2
```

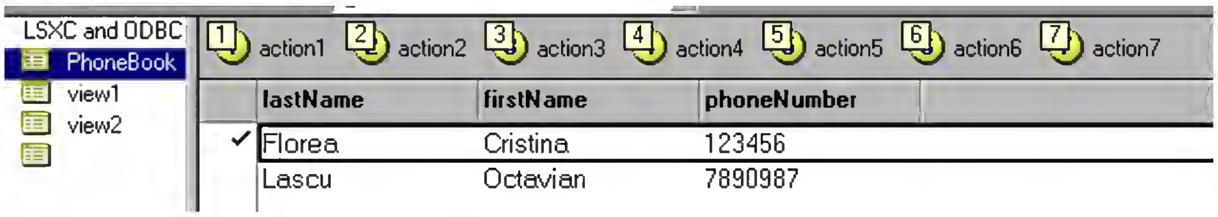
In order to run **ACTION5** do the following step:

- ✓ Open the view PhoneBook, and push onto **ACTION5.**

In order to run **ACTION6** do the following steps:

- ✓ Select LSXCODBC.NSF DataBase.
- ✓ Open the view PhoneBook, select the document **Florea Costica 123456** , open it in edit mode and instead of **Costica**, put **Cristina** and save the document.
- ✓ Open the view PhoneBook, select the document **Florea Cristina 123456** and push onto ACTION6.

The result is as follows:



lastName	firstName	phoneNumber
Florea	Cristina	123456
Lascu	Octavian	7890987

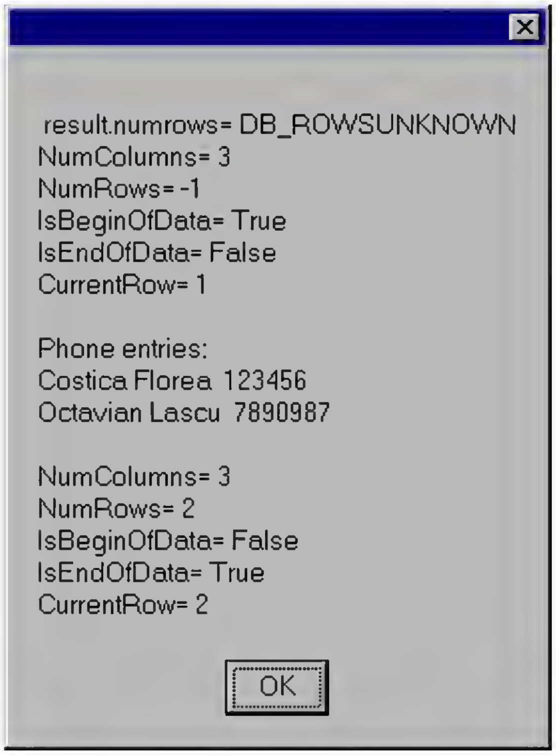
The Content of Phone table in SAMPLE DataBase:

```
LASTNAME        FIRSTNAME        PHONENO
-----------------------------------------------------------
Florea          Cristina         123456
Lascu           Octavian         7890987
```

In order to run **ACTION7** do the following step:

✓ Open the view PhoneBook, and push onto ACTION7.

The result is as follows:



```
result.numrows= DB_ROWSUNKNOWN
NumColumns= 3
NumRows= -1
IsBeginOfData= True
IsEndOfData= False
CurrentRow= 1

Phone entries:
Costica Florea  123456
Octavian Lascu  7890987

NumColumns= 3
NumRows= 2
IsBeginOfData= False
IsEndOfData= True
CurrentRow= 2
```

In order to run **AGENT11** do the following step:

✓ Select LSXCODBC.NSF DataBase ---> Actions ---> AGENT11

Example 3.11

In this example there is the form FORM2 that contains two fields (text + editable) named dataSource and Table, four buttons named “Data Source”, “Table”, Postopen”, “QueryClose”, and two actions named “List Fields” and “List Procedure”.

The button “Postopen” sets the objects, gets the names of the available data sources, writes the first one to the dataSource field, gets the names of the tables for the data source and writes the first one to the Table field.

The button “Data Source” writes the name of the next data source to the dataSource field, gets the tables for the new data source and writes the first one to the Table field.

The button “Table” writes the name of the next table to the Table field.

The action “List Fields” displays the names of all the fields for the current data source and table.

The action “List Procedures” displays the name of all the procedures for the current data source.

In order to achieve this objective do the following steps:

Step A - 3.11

Create the form FORM2 having:

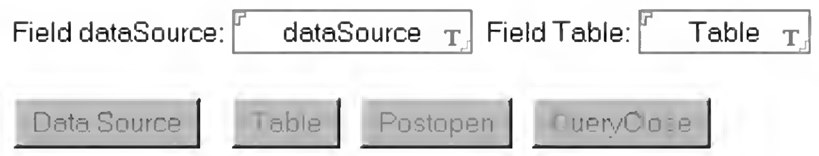
Globals->Options:

```
Option Public
USELSX "*LSXODBC"
```

Global->Declarations:

```
Dim con As odbcconnection
Dim datasources As Variant
Dim tables As Variant
Dim thisdsn As Integer
Dim thistable As Integer
Dim workspace As notesuiworkspace
Dim uidoc As notesuidocument
Dim parola As String
Dim userid1 As String
```

The image of FORM2 is as follows:



Step B - 3.11

Create the following LotusScript for the button **Data Source**:

```
Sub Click(Source As Button)
        userid1="Administrator"
        parola="rac4you"
        If thisdsn=Ubound(datasources) Then
                thisdsn=Lbound(datasources)
        Else
                thisdsn=thisdsn+1
        End If
        Call uidoc.fieldsettext("dataSource", datasources(thisdsn))
        If (datasources(thisdsn)="SAMPLE") Then
                tables=con.listtables(datasources(thisdsn),userid1,parola)
                If Ubound(tables) <> 0 Then
                        thistable=Lbound(tables)
                        Call uidoc.fieldsettext("Table",tables(thistable))
                End If
        Else
                Call uidoc.fieldsettext("Table","")
        End If
End Sub
```

Step C - 3.11

Create the following LotusScript for the button **Table**:

```
Sub Click(Source As Button)
        If (datasources(thisdsn)="SAMPLE") Then
                If Ubound(tables) <>0 Then
                        userid1="Administrator"
                        parola="rac4you"
                        If thistable=Ubound(tables) Then
                                thistable=Lbound(tables)
                        Else
                                thistable=thistable+1
                        End If
                        Call uidoc.fieldsettext("Table", tables(thistable))
                End If
        End If
End Sub
```

Step D - 3.11

Create the following LotusScript for the button **Postopen**:

```
Sub Click(Source As Button)
        userid1="Administrator"
        parola="rac4you"
        Set workspace=New notesuiworkspace
        Set uidoc=workspace.currentdocument
        Set con=New odbcconnection
        con.silentmode=True
        datasources=con.listdatasources
        thisdsn=Lbound(datasources)
        Call uidoc.fieldsettext("dataSource", datasources(thisdsn))
        If (datasources(thisdsn)="SAMPLE") Then
                tables=con.listtables(datasources(thisdsn),userid1,parola)
                If Ubound(tables) <> 0 Then
                        thistable=Lbound(tables)
                        Call uidoc.fieldsettext("Table",tables(thistable))
                End If
        Else
                Call uidoc.fieldsettext("Table","")
        End If
End Sub
```

Step E - 3.11

Create the following LotusScript for the button **QueryClose**:

```
Sub Click(Source As Button)
        If con.isconnected Then
                con.disconnect
        End If
End Sub
```

Step F - 3.11

Create the following LotusScript for the action **List Fields**:

```
Sub Click(Source As Button)
        If (thisdsn<>0) And (thistable<>0 ) Then
                If (datasources(thisdsn)="SAMPLE") Then
                        userid1="Administrator"
                        parola="rac4you"
                        Dim msg As String
                        Dim fields As Variant
                        Call con.connectto(datasources(thisdsn),userid1,parola)
                        If con.isconnected Then
                                fields=con.listfields(tables(thistable))
                                If Ubound(fields) <> 0 Then
                                        msg=tables(thistable) & " contains the following fields: " & Chr(10)
                                        For o%=Lbound(fields) To Ubound(fields)
                                                msg=msg & Chr(10) & fields(o%)
                                        Next
                                        Messagebox msg & " " & Chr(10) & Chr(10) & "for " & con.datasourcename
& " DataBase"
                                Else
                                        Messagebox "No fields in " & tables(thistable) & " of " & con.datasourcename
& " DataBase"
                                End If
                                con.disconnect
                        End If
                Else
                        Messagebox "This is not SAMPLE DataBase"
                End If
        Else
                Messagebox "The operation is not accepted"
        End If
End Sub
```

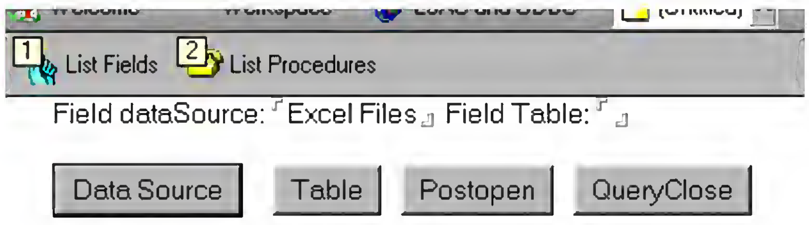
Step G - 3.11

Create the following LotusScript for the action **List Procedures**:

```
Sub Click(Source As Button)
        If (thisdsn<>0) And (thistable<>0 ) Then
                If (datasources(thisdsn)="SAMPLE") Then
                        userid1="Administrator"
                        parola="rac4you"
                        Dim msg As String
                        Dim procs As Variant
                        Call con.connectto(datasources(thisdsn),userid1,parola)
                        If con.isconnected Then
                                procs=con.listprocedures
                                If Ubound(procs) <> 0 Then
                                        msg=con.datasourcename & " DataBase contains the following procedures: " &
Chr(10)
                                        For o%=Lbound(procs) To Ubound(procs)
                                                msg=msg & Chr(10) & procs(o%)
                                        Next
                                        Messagebox msg & " " & Chr(10) & Chr(10) & "for " & con.datasourcename
& " DataBase"
                                Else
                                        Messagebox "No procedures for " & con.datasourcename & " DataBase"
                                End If
                                con.disconnect
                        End If
                Else
                        Messagebox "This is not SAMPLE DataBase"
                End If
        Else
                Messagebox "The operation is not accepted"
        End If
End Sub
```

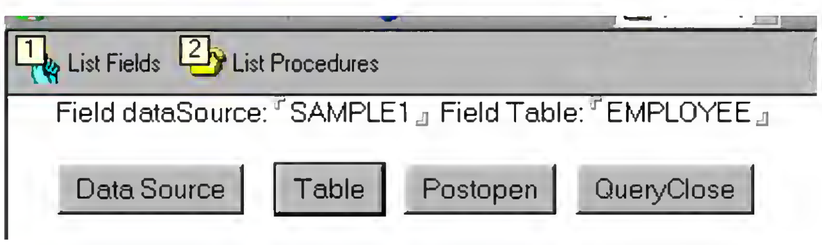
In order to run **EXAMPLE 3.11** do the following steps:

✓ Select LSXCODBC.NSF DataBase ---> Create ---> FORM2
✓ Push onto the button **Postopen. It is essential that this be done first !**
✓ If you push onto the button **Data Source,** you get the following information:



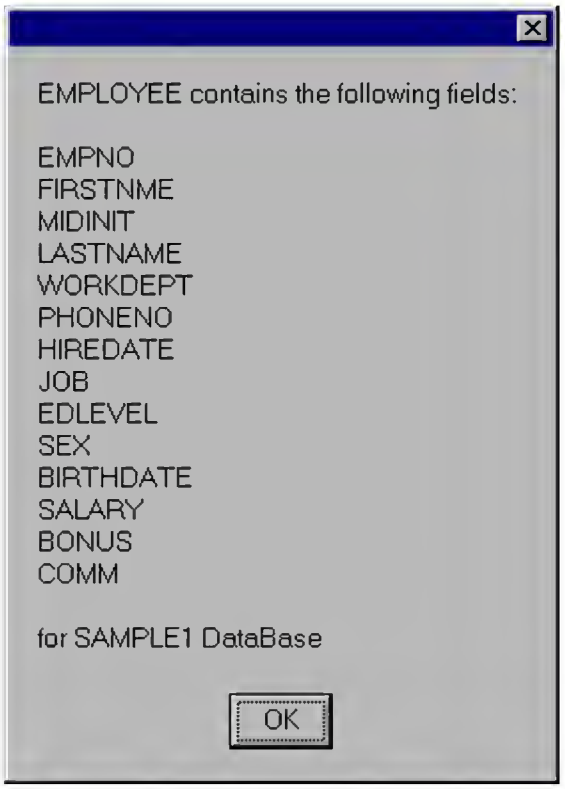
Field dataSource: ⌜Excel Files⌟ Field Table: ⌜ ⌟

✓ If you push onto the button **Table**, you get the following information:



Field dataSource: SAMPLE1 Field Table: EMPLOYEE

✓ If you push onto the action **List Fields**, you get the following information:



EMPLOYEE contains the following fields:

```
EMPNO
FIRSTNME
MIDINIT
LASTNAME
WORKDEPT
PHONENO
HIREDATE
JOB
EDLEVEL
SEX
BIRTHDATE
SALARY
BONUS
COMM
```

for SAMPLE1 DataBase

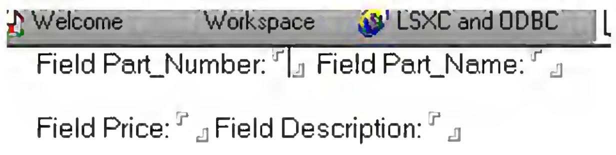
Example 3.12

In this example, each time when you exit from the field Part_Number (inside of which you must type a valid serial number taken from EMPNO of EMPLOYEE table), the code associated with this field, automatically fills in the fields Part_Name (with the value of FIRSTNME), Price (with the value of LASTNAME), Description (with the value of WORKDEPT).

In order to achieve this objective do the following steps:

Step A - 3.12

Create the form FORM3 having the following fields (text + editable): Part_Number, Part_Name, Price, Description.



Field Part_Number: ⌜⌟ Field Part_Name: ⌜⌟

Field Price: ⌜⌟ Field Description: ⌜⌟

Step B - 3.12

Create the following LotusScript code for the field Part_Number:

```
Sub Exiting(Source As Field)
        Dim con As New odbcconnection
        Dim qry As New odbcquery
        Dim res As New odbcresultset
        Dim ws As New notesuiworkspace
        Dim uidoc As notesuidocument
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        Set uidoc=ws.currentdocument
        Call con.disconnect
        If con.connectto(dsn,userid, parola) Then
                Set qry.connection=con
                qry.sql="Select * from EMPLOYEE where EMPNO= '"+uidoc.fieldgettext("Part_Number") + "'"
                Set res.query=qry
                res.execute
                If res.isresultsetavailable=True Then
                        res.firstrow
                        Call uidoc.fieldsettext("Part_Number",res.getvalue("EMPNO"))
                        Call uidoc.fieldsettext("Part_Name",res.getvalue("FIRSTNME"))
                        Call uidoc.fieldsettext("Price",res.getvalue("LASTNAME"))
                        Call uidoc.fieldsettext("Description",res.getvalue("WORKDEPT"))
                Else
```

```
                        Messagebox "No Information found for " & uidoc.fieldgettext("Part_Number")
                End If
                res.close(DB_CLOSE)
                con.disconnect
        Else
                Messagebox "Could not connect to " & dsn & " DataBase"
        End If
End Sub
```

In order to run **Example 3.12** do the following steps:

✓ Select LSXCODBC.NSF DataBase ---> Create ---> FORM3
✓ Fill in the field Part_Number with an EMPNO value, let say 000020
✓ Exit from the field Part_Number trying to get into the field Part_Name. After a while, you'll see, the fields Part_Name, Price, Description are automatically filled with values taken from the table EMPLOYEE for that EMPNO(000020).

The result is as follows:

```
Field Part_Number: 000020  Field Part_Name: MICHAEL

Field Price: THOMPSON Field Description: B01
```

Example 3.13

In order to understand this example, read the paragraph **"Tips and techniques - Handling an ODBC event"** from the book **Domino Release 5. Domino Designer Programming Guide, Volume 2.**

In this example, the values of a row in an ODBC table are displayed as fields in FORM4. The user can use buttons to get the next and previous rows. The event handler **AfterPositionChange** displays the number of the current row in another field on the form FORM4.

In order to achieve this objective do the following steps:

Step A - 3.13

Create the form FORM4 having:
The fields (text + editable) empno, lastname, hiredate, RowNumber.
The buttons: "Postopen", "Get the Next Row", "Get the Previous Row", "QueryClose" and the following features:

Globals->Options:

```
Option Public
USELSX "*LSXODBC"
```

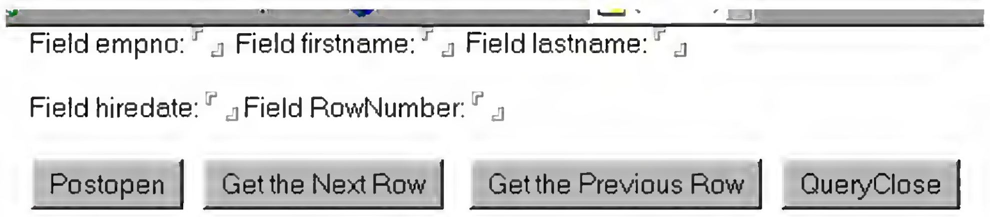
Global->Declarations:

```
Dim con As odbcconnection
Dim qry As odbcquery
Dim result As odbcresultset
Dim msg As String
Dim dsn As String
Dim userid1 As String
Dim parola As String
```

Global->afterpositionchange(res as odbcresultset)

```
Sub afterpositionchange(res As odbcresultset)
        Dim ws As New notesuiworkspace
        Dim source As notesuidocument
        Set source=ws.currentdocument
        Call source.fieldsettext("RowNumber",Cstr(res.currentrow))
End Sub
```

The image of FORM4 is as follows:



Step B - 3.13

Create the following LotusScript for the button **Postopen**:

```
Sub Click(Source As Button)
        dsn="SAMPLE"
        userid1="Administrator"
        parola="rac4you"
        Set con=New odbcconnection
        Set qry=New odbcquery
        Set result=New odbcresultset
        Set qry.connection=con
        Set result.query=qry
        Call con.disconnect
        Call con.connectto(dsn,userid1,parola)
        If Not con.isconnected Then
                msg="Could not connect to " & dsn & " DataBase" & Chr(10)
                If con.geterror <> DBstsSUCCESS Then
                        msg=msg & "ExtendedErrorMessage= " & con.getextendederrormessage
                        msg=msg & " Error= " & con.geterror & " ErrorMessage= " & con.geterrormessage
                End If
                Messagebox msg
                End
        End If
        On Event afterfirstrow From result Call afterpositionchange
        On Event afterlastrow From result Call afterpositionchange
        On Event afternextrow From result Call afterpositionchange
        On Event afterprevrow From result Call afterpositionchange
        qry.sql="select * from EMPLOYEE order by LASTNAME"
        result.execute
        Dim ws As New notesuiworkspace
        Dim source1 As notesuidocument
        Set source1=ws.currentdocument
        If Not source1.editmode Then
                source1.editmode=True
        End If
        result.firstrow
        Call source1.fieldsettext("empno",Cstr(result.getvalue("EMPNO")))
        Call source1.fieldsettext("firstname",Cstr(result.getvalue("FIRSTNME")))
        Call source1.fieldsettext("lastname",Cstr(result.getvalue("LASTNAME")))
        Call source1.fieldsettext("hiredate",Cstr(result.getvalue("HIREDATE")))
End Sub
```

Step C - 3.13

Create the following LotusScript for the button **Get the Next Row**:

```
Sub Click(Source As Button)
        Dim workspace As New notesuiworkspace
        Dim uidoc As notesuidocument
        Set uidoc=workspace.currentdocument
        If Not result.isendofdata Then result.nextrow
        Call uidoc.fieldsettext("empno", Cstr(result.getvalue("EMPNO")))
        Call uidoc.fieldsettext("firstname", Cstr(result.getvalue("FIRSTNME")))
        Call uidoc.fieldsettext("lastname", Cstr(result.getvalue("LASTNAME")))
        Call uidoc.fieldsettext("hiredate", Cstr(result.getvalue("HIREDATE")))
End Sub
```

Step D - 3.13

Create the following LotusScript for the button **Get the Previous Row**:

```
Sub Click(Source As Button)
        Dim workspace As New notesuiworkspace
        Dim uidoc As notesuidocument
        Set uidoc=workspace.currentdocument
        If Not result.isendofdata Then result.prevrow
        Call uidoc.fieldsettext("empno", Cstr(result.getvalue("EMPNO")))
        Call uidoc.fieldsettext("firstname", Cstr(result.getvalue("FIRSTNME")))
        Call uidoc.fieldsettext("lastname", Cstr(result.getvalue("LASTNAME")))
        Call uidoc.fieldsettext("hiredate", Cstr(result.getvalue("HIREDATE")))
End Sub
```

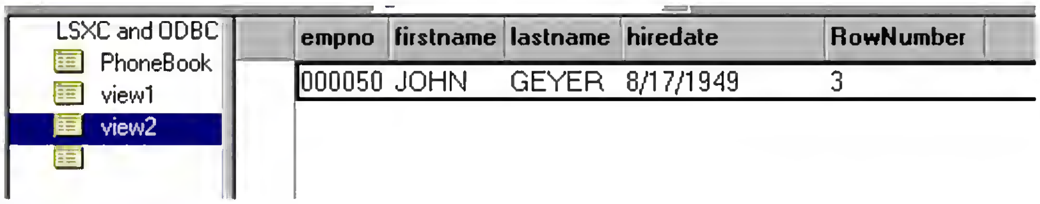
Step E - 3.13

Create the following LotusScript for the button **QueryClose**:

```
Sub Click(Source As Button)
        result.close(DB_CLOSE)
        If result.geterror <> DBstsSUCCESS Then
                msg="ExtendedErrorMessage= " & result.getextendederrormessage
                msg=msg & " Error= " & result.geterror & " ErrorMessage= " & result.geterrormessage
                Messagebox msg
        End If
        con.disconnect
End Sub
```

In order to run **EXAMPLE 3.13** do the following steps:

- ✓ Select LSXCODBC.NSF DataBase ---> Create ---> FORM4
- ✓ Push onto the button **Postopen. This action is mandatory to be the first one.**
- ✓ If you push onto the buttons **Get the Next Row / Get the Previous Row** and save the document, you get something like the following information:



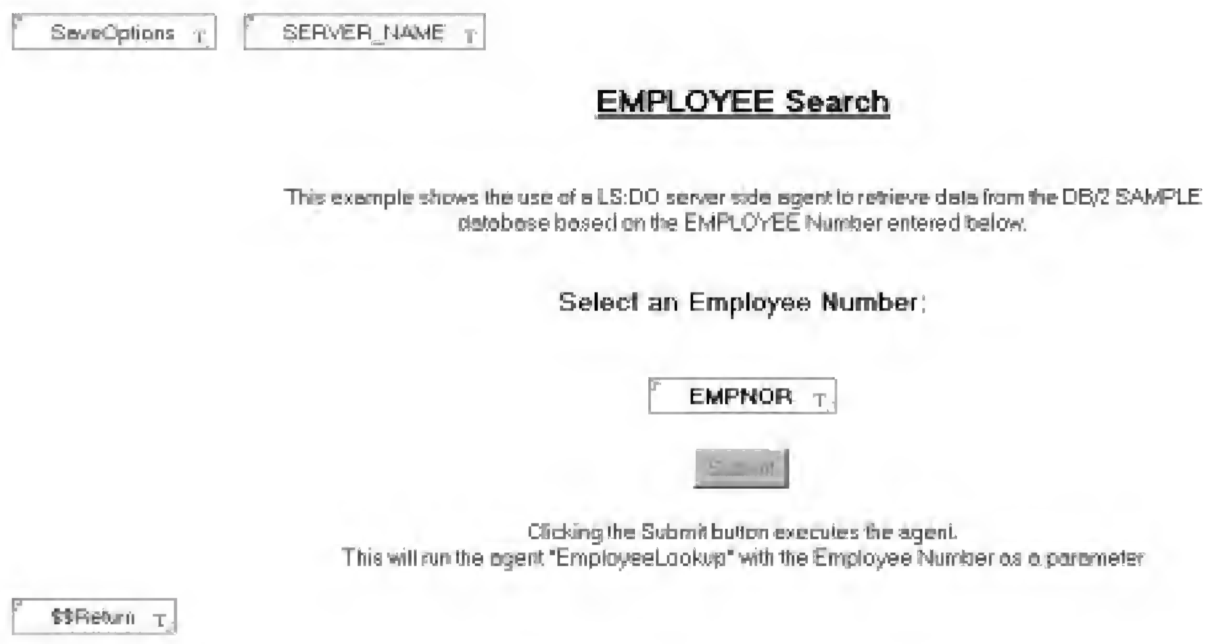
empno	firstname	lastname	hiredate	RowNumber
000050	JOHN	GEYER	8/17/1949	3

Example 3.14

This example shows how to access external databases via a Web browser and Domino Server, using ODBC. To access the data from the Web browser, you must define an ODBC connection to external data source and must write the ODBC code in an agent that runs via a URL command.The display of the data needed to be formatted in HTML. In this example, giving the employee's serial number, we get information about an employee from SAMPLE database. **Example 3.14 is similar with Example 2.21; the only difference is that Example 2.21 uses LSX LC and Example 3.14 uses ODBC.**
In order to execute Example 3.14 do the following steps:

Step A - 3.14

Create a form on LSXCODBC.NSF, named FORM5 having the following structure:


SaveOptions SERVER_NAME

EMPLOYEE Search

This example shows the use of a LS:DO server side agent to retrieve data from the DB/2 SAMPLE database based on the EMPLOYEE Number entered below.

Select an Employee Number:

EMPNOR

Submit

Clicking the Submit button executes the agent.
This will run the agent "EmployeeLookup" with the Employee Number as a parameter

$$Return

Let's detail the above form:

- Field SaveOptions: text + computed, formula: "0"
- Field SERVER_NAME: text + computed, formula: SERVER_NAME
- Field EMPNOR: text + editable
- Field $$Return: text + computed, formula:

```
@Return("[http://"+SERVER_NAME+"/"+@ReplaceSubstring(@Subset(@DbName;-1);"\\";"/")+"/EmployeeLookup?OpenAgent&"+EMPNOR+"]")
```

- Button Submit: JavaScript Formula: this.form.submit()

*

*   *

The Fields: SaveOptions, SERVER_NAME, $$Return have in "Paragraph Hide When":
Hide paragraph from:
* Notes R4.6 or later   * Web browser
Hide paragraph when document is:
* opened for reading   * opened for editing

* printed

<u>Step B- 3.14</u>

Create the agent named **EmployeeLookup** having the features: Share Agent + Manually from agent list + Should act on all documents in database.

Create the following LotusScript code for agent **EmployeeLookup**:

```
Option Public
Uselsxc "*LSXODBC"

Sub Initialize
	Dim session As New  notessession
	Dim doc As notesdocument
	Dim conn As New odbcconnection
	Dim query As  New odbcquery
	Dim data As New odbcresultset
	Dim var1 As Integer
	Set query.connection=conn
	Set data.query=query
	Set doc=session.documentcontext
	Set db=session.currentdatabase
	conn.silentmode=True
	Dim dsn As String
	Dim userid As String
	Dim parola As String
	dsn="SAMPLE"
	userid="Administrator"
	parola="rac4you"
	urlstring=doc.Query_String(0)
	urllength=Len(urlstring)
	paramposition=Instr(urlstring,"&")+1
	webparam=Mid(urlstring,paramposition,urllength-paramposition+1)
	Call conn.disconnect
	If Not conn.connectto(dsn,userid,parola) Then
		Print "Not OK, Could not connect to " & dsn & " DataBase."
		error%=conn.geterror
		message$=conn.geterrormessage
		extendedmessage$=conn.getextendederrormessage
		Print message$ & "<br>"
		Print "Error Code: " & Str$(error%)
		Print "Extended Error: " & extendedmessage$ & "<hr>"
		End
	End If
```

```
        query.sql="select * from EMPLOYEE where EMPNO='" & webparam & "'"
        If Not data.execute Then
                Print "Not OK, Could not Select from " & dsn & " DataBase !"
                error%=conn.geterror
                message$=conn.geterrormessage
                extendedmessage$=conn.getextendederrormessage
                Print message$ & "<br>"
                Print "Error Code: " & Str$(error%)
                Print "Extended Error: " & extendedmessage$ & "<hr>"
                End
        End If
        var1=0
        Do
                data.nextrow
                empno=data.getvalue("EMPNO",empno)
                If empno=webparam Then
                        var1=1
                        firstnme=data.getvalue("FIRSTNME",firstnme)
                        midinit=data.getvalue("MIDINIT",midinit)
                        lastname=data.getvalue("LASTNAME",lastname)
                        workdept=data.getvalue("WORKDEPT",workdept)
                        phoneno=data.getvalue("PHONENO",phoneno)
                        hiredate=data.getvalue("HIREDATE",hiredate)
                        job=data.getvalue("JOB",job)
                        edlevel=data.getvalue("EDLEVEL",edlevel)
                        sex=data.getvalue("SEX",sex)
                        birthdate=data.getvalue("BIRTHDATE",birthdate)
                        salary=data.getvalue("SALARY",salary)
                        bonus=data.getvalue("BONUS",bonus)
                        comm=data.getvalue("COMM",comm)
                        Print "<head><body>"
                        Print "<h3>This is the information for employee: " & webparam & "</h3>"
                        Print "EMPNO: " & empno & "<br>"
                        Print "FIRSTNAME: " & firstnme & "<br>"
                        Print "MIDINIT: " & midinit & "<br>"
                        Print "LASTNAME: " & lastname & "<br>"
                        Print "<br>"
                        Print "WORKDEPT: <a href=./DeptLookup?OpenAgent&" & workdept & ">" & workdept
& "</a>" & "<br>"
                        Print "PHONENO: " & phoneno & "<br>"
                        Print "HIREDATE: " & hiredate & "<br>"
                        Print "JOB: " & job & "<br>"
                        Print "EDLEVEL: " & edlevel & "<br>"
                        Print "SEX: " & sex & "<br>"
                        Print "BIRTHDATE: " & birthdate & "<br>"
                        Print "SALARY: " & salary & "<br>"
                        Print "BONUS: " & bonus & "<br>"
                        Print "COMM: " & comm & "<br>"
                        Print "<br><br>"
                        Print "Thank You"
                End If
        Loop Until data.isendofdata
        If var1 <> 1 Then
                Print "Not OK, The EMPLOYEE ID cannot be found in " & dsn & " DataBase !"
                error%=query.geterror
                message$=query.geterrormessage
                extendedmessage$=query.getextendederrormessage
                Print message$ & "<br>"
                Print "Error Code: " & Str$(error%)
                Print "Extended Error: " & extendedmessage$ & "<hr>"
                End
        End If
        data.close(DB_CLOSE)
        conn.disconnect
End Sub
```

<u>Step C- 3.14</u>

Create the agent named **DeptLookup** having the features: Share Agent + Run once(@command may be used).

Create the following LotusScript code for agent **DeptLookup**:

```
Option Public
Uselsxc "*LSXODBC"

Sub Initialize
        Dim session As New  notessession
        Dim doc As notesdocument
        Dim conn As New odbcconnection
        Dim query As  New odbcquery
        Dim data As New odbcresultset
        Set query.connection=conn
        Set data.query=query
        Set doc=session.documentcontext
        Set db=session.currentdatabase
        conn.silentmode=True
        Dim dsn As String
        Dim userid As String
        Dim parola As String
        dsn="SAMPLE"
        userid="Administrator"
        parola="rac4you"
        urlstring=doc.Query_String(0)
        urllength=Len(urlstring)
        paramposition=Instr(urlstring,"&")+1
        webparam=Mid(urlstring,paramposition,urllength-paramposition+1)
        Call conn.disconnect
        If Not conn.connectto(dsn,userid,parola) Then
                Print "Not OK, Could not connect to " & dsn & " DataBase."
                error%=conn.geterror
                message$=conn.geterrormessage
                extendedmessage$=conn.getextendederrormessage
                Print message$ & "<br>"
                Print "Error Code: " & Str$(error%)
                Print "Extended Error: " & extendedmessage$ & "<hr>"
                End
        End If
        query.sql="select * from EMPLOYEE where WORKDEPT='" & webparam & "'"
        If Not data.execute Then
                Print "Not OK, Could not Select from " & dsn & " DataBase !"
                error%=conn.geterror
                message$=conn.geterrormessage
                extendedmessage$=conn.getextendederrormessage
                Print message$ & "<br>"
                Print "Error Code: " & Str$(error%)
                Print "Extended Error: " & extendedmessage$ & "<hr>"
                End
```

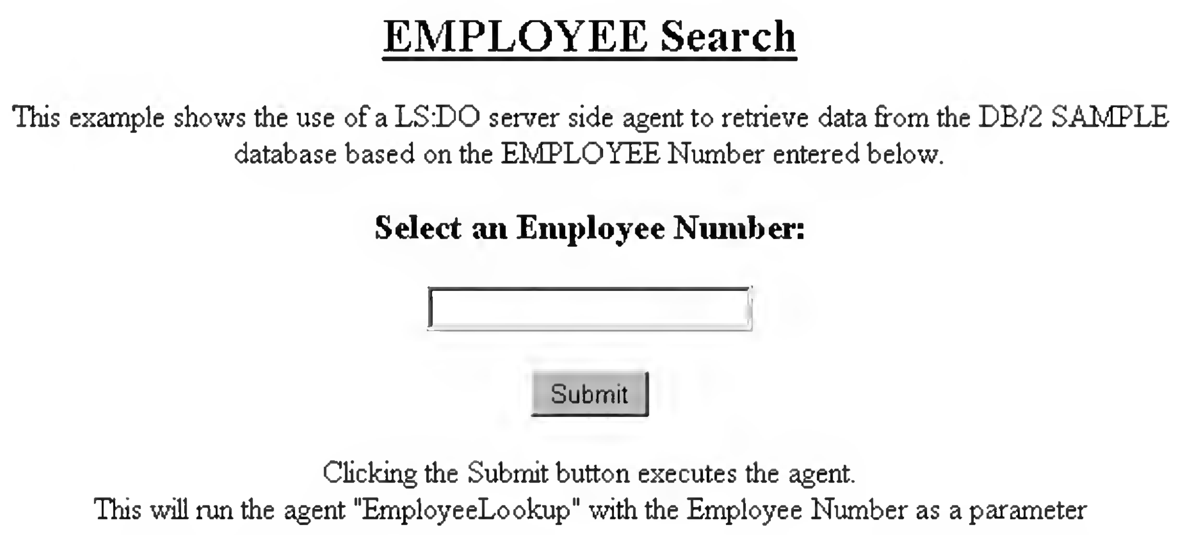
```
    End If
    Print "<head><body>"
    Print "<h3>These are other employees that work in department " & webparam & "</h3>"
    Print "<table border="1">"
    Print "<tr>"
    Print "<td>EMPNO</td>"
    Print "<td>FIRSTNME</td>"
    Print "<td>MIDINIT</td>"
    Print "<td>LASTNAME</td>"
    Print "<td>PHONENO</td>"
    Print "<td>HIREDATE</td>"
    Print "<td>JOB</td>"
    Print "<td>EDLEVEL</td>"
    Print "<td>SEX</td>"
    Print "<td>BIRTHDATE</td>"
    Print "<td>SALARY</td>"
    Print "<td>BONUS</td>"
    Print "<td>COMM</td>"
    Print "<tr>"
    Do
        data.nextrow
        empno=data.getvalue("EMPNO",empno)
        firstnme=data.getvalue("FIRSTNME",firstnme)
        midinit=data.getvalue("MIDINIT",midinit)
        lastname=data.getvalue("LASTNAME",lastname)
        phoneno=data.getvalue("PHONENO",phoneno)
        hiredate=data.getvalue("HIREDATE",hiredate)
        job=data.getvalue("JOB",job)
        edlevel=data.getvalue("EDLEVEL",edlevel)
        sex=data.getvalue("SEX",sex)
        birthdate=data.getvalue("BIRTHDATE",birthdate)
        salary=data.getvalue("SALARY",salary)
        bonus=data.getvalue("BONUS",bonus)
        comm=data.getvalue("COMM",comm)
        Print "<tr>"
        Print "<td>" & empno & "</tr>"
        Print "<td><a href=./EmployeeLookup?OpenAgent&" & empno & ">" & firstnme & "</a>" &
"</tr>"
        Print "<td>" & midinit & "</tr>"
        Print "<td>" & lastname & "</tr>"
        Print "<td>" & phoneno & "</tr>"
        Print "<td>" & hiredate & "</tr>"
        Print "<td>" & job & "</tr>"
        Print "<td>" & edlevel & "</tr>"
        Print "<td>" & sex & "</tr>"
        Print "<td>" & birthdate & "</tr>"
        Print "<td>" & salary & "</tr>"
        Print "<td>" & bonus & "</tr>"
        Print "<td>" & comm & "</tr>"
        Print "</tr>"
        Print "</br>"
    Loop Until data.isendofdata
    Print "</table>"
    Print "</body></head>"
    data.close(DB_CLOSE)
    conn.disconnect
End Sub
```

In order to run **Example 3.14** do the following steps:

✓ Open a Web browser and type the following URL:

http://mummer.ism.can.ibm.com/test1/lsxcodbc.nsf/form5

The result on the Web browser is as follows:

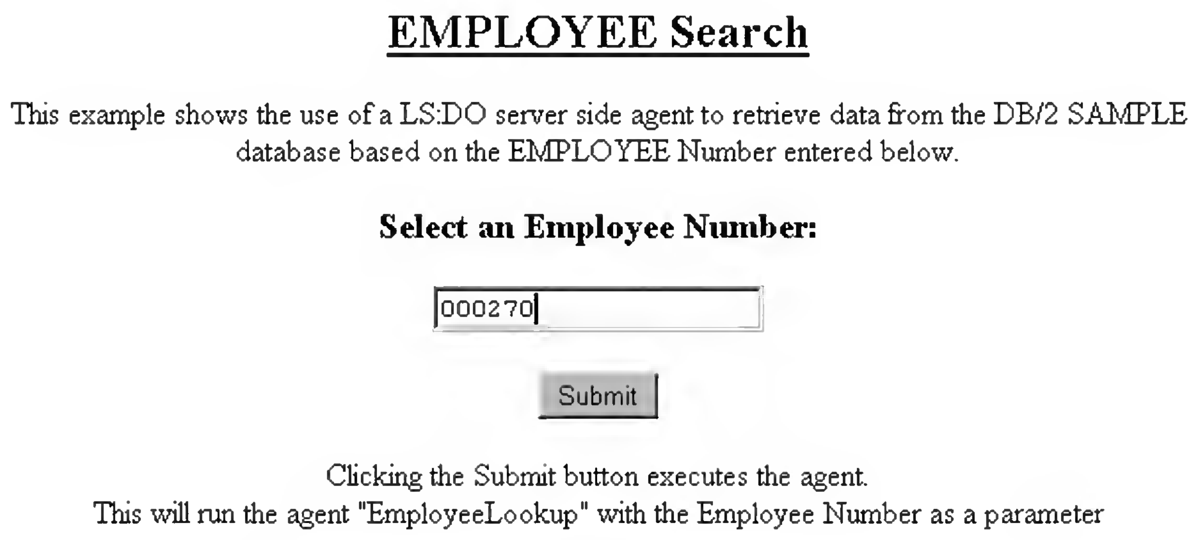

EMPLOYEE Search

This example shows the use of a LS:DO server side agent to retrieve data from the DB/2 SAMPLE database based on the EMPLOYEE Number entered below.

Select an Employee Number:

Clicking the Submit button executes the agent.
This will run the agent "EmployeeLookup" with the Employee Number as a parameter

✓ Type the following Serial Number: **000270** and Click onto Submit button when finished.


EMPLOYEE Search

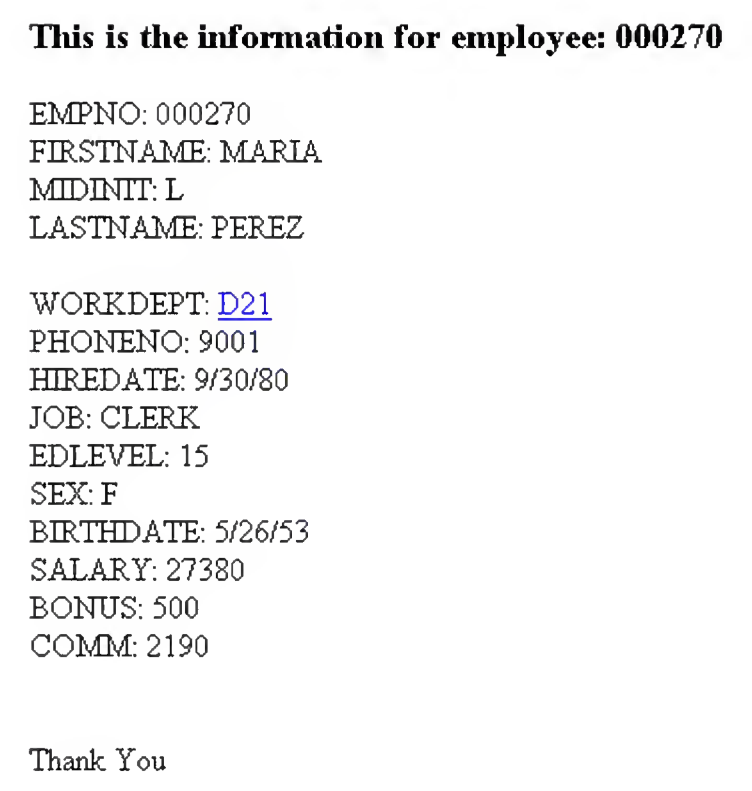
This example shows the use of a LS:DO server side agent to retrieve data from the DB/2 SAMPLE database based on the EMPLOYEE Number entered below.

Select an Employee Number:

000270

Clicking the Submit button executes the agent.
This will run the agent "EmployeeLookup" with the Employee Number as a parameter

After a while the Web browser brings up the following information:



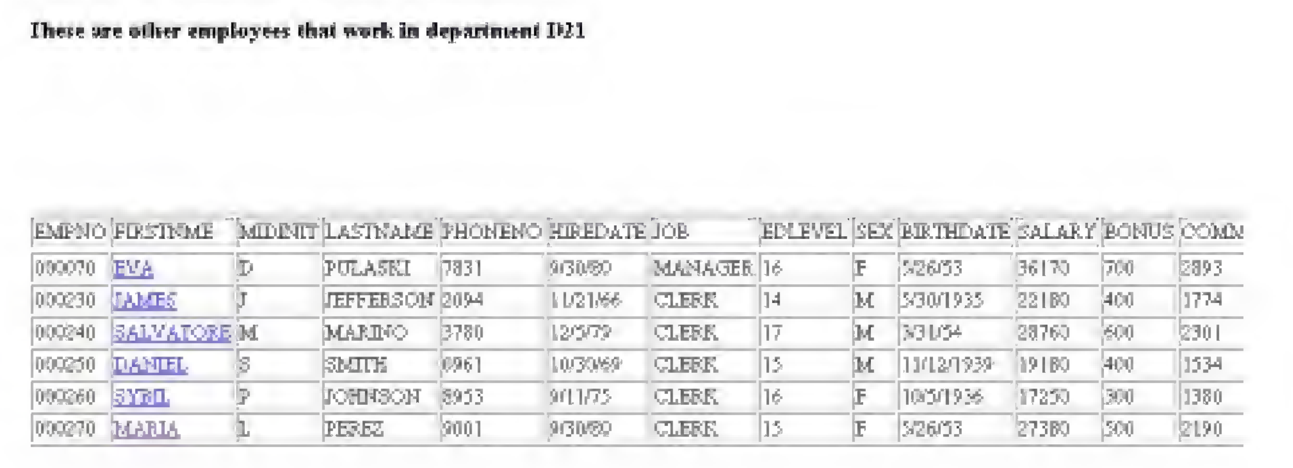
This is the information for employee: 000270

EMPNO: 000270
FIRSTNAME: MARIA
MIDINIT: L
LASTNAME: PEREZ

WORKDEPT: D21
PHONENO: 9001
HIREDATE: 9/30/80
JOB: CLERK
EDLEVEL: 15
SEX: F
BIRTHDATE: 5/26/53
SALARY: 27380
BONUS: 500
COMM: 2190

Thank You

✓ Click on **D21** Reference Link in order to see what other persons work in the same department.



These are other employees that work in department D21

EMPNO	FIRSTNME	MIDINIT	LASTNAME	PHONENO	HIREDATE	JOB	EDLEVEL	SEX	BIRTHDATE	SALARY	BONUS	COMM
000070	EVA	D	PULASKI	7831	9/30/80	MANAGER	16	F	5/26/53	36170	700	2893
000230	JAMES	J	JEFFERSON	2094	11/21/66	CLERK	14	M	5/30/1935	22180	400	1774
000240	SALVATORE	M	MARINO	3780	12/5/79	CLERK	17	M	3/31/54	28760	600	2301
000250	DANIEL	S	SMITH	0961	10/30/69	CLERK	15	M	11/12/1939	19180	400	1534
000260	SYBIL	P	JOHNSON	8953	9/11/75	CLERK	16	F	10/5/1936	17250	300	1380
000270	MARIA	L	PEREZ	9001	9/30/80	CLERK	15	F	5/26/53	27380	500	2190

✓ Click on any Name, listed under column FIRSTNME. Actually behind each name is a Reference Link. After a while the Web browser brings up the information for that specific Name in the same format as for **Maria Perez**: **This is the information for employee .....**

You can play around selecting a lot of EMPNOs and FIRSTNMEs from EMPLOYEE table.